Don Lancaster's

Ask the Guru

Selected reprints — volume II
Computer Shopper series (November 1987 - December 1989)

Box 809, Thatcher, AZ 85552 (520) 428-4073

Book-on-Demand self-published using the Apple IIe computer
and the LaserWriter NTX. All graphics were done in their
entirety by ProDOS AppleWriter 2.1.

Introduction

Hmmm It seems that we somehow made it to Volume II. Anyway, welcome back to our ongoing series of reprints from my *Ask the Guru* column originally found in *Computer Shopper* magazine. This volume will begin with column #32 which first appeared in December of 1987.

What is this all about? Darned if I know. Be sure to let me know if you ever find out. In theory, we are supposed to deal with stuff like Apple IIe computing, laser printing, *PostScript* programming, and the insider desktop publishing secrets. But most anything at all can and will come up. As before, I go on the assumption that if I am interested enough in something to get involved with it, then others like yourself may also be.

These volumes are also an ongoing experiment in *book on demand* PostScript laser printing. What you have here was literally beat out on a brick in my back yard. All of the figures, all of the artwork, and all of the text was done by using the *AppleWriter* word processor on an Apple IIe, and on-demand printed, one self-collating custom copy at a time, on an Apple LaserWriter. Thanks to some ultra-sneaky programming and comm tricks, this all happens at the "wide open" LaserWriter print speed. The page makeready time is zero for most pages, a trick that is trivially easy on the Apple IIe, yet quite difficult on a Mac II or a 386 clone.

Because of all the toner reloading tricks you will read about here, my LaserWriter per-page toner costs are one-fifteenth of the going rate. Which helps make my laser printing economics more than cost competitive with jiffy printing. Binding is via a *Unibind* toaster, and shearing by way of "borrowed" time on a thirty party shear that I keep sharp for them.

My ongoing thanks to *Computer Shopper* editor Stan Veit for letting me say what I want to say when and how I want to say it. Should you want to subscribe to *Computer Shopper,* do give them at call at (407) 269-3211. ✦

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 now the author of 23 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. He is considered by others to be the patron saint of the Walter Mitties of the world. And, he is considered by yet others to be the . . . er, better skip that one.

His monthly columns include both the *Ask the Guru* and *LaserWriter Corner* over in *Computer Shopper*, and his *Hardware Hacker* column in *Radio Electronics* magazine.

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*, the *Active Filter Cookbook*, *Apple Assembly Cookbook*, his *Hardware Hacker* reprints, *Don Lancaster's PostScript Secrets*, and his *Intro to PostScript* video.

Don's current software offerings include his *PostScript Show and Tell*, and *PostScript Work in Progress*, plus a few companion disks for his various books.

Don is the head honcho of *Synergetics*, a new-age design and consulting firm that specializes in Apple computing, laser printing, *PostScript* program utilities, electronic prototyping, book-on-demand publishing, technical writing, and innovative software design. His avocations include firefighting, cave exploration, bicycling, and, of course, *tinaja* questing.

Don maintains a no charge voice helpline at (520) 428-4073. He welcomes your calls and letters. Best calling times are 8-5 weekdays, *Mountain Standard Time.* ✦

Table of Contents

more . . .

Don Lancaster's

ASK THE GURU

November, 1987

The paper cutters
Patching programs
Rubbergrid utilities
New LaserWriter ROM
PostScript fundamentals

Apple Computer desktop publishing introductions so far this fall have seemed rather low key, at least to this writing. In mid-September, they quietly announced a new LaserWriter upgrade to those Version 47 ROM's and adjusted their list price so it no longer is an outright joke.

The V47 ROM's will execute most routines 33 percent faster. They also eliminate some (but not all) of the nasty bugs in the older Version 38.

A list of all the major LaserWriter Version 38 bugs appears in figure one, along with some of the workarounds. Note that *all* of the recent LaserWriter's had Version 38 ROM's in them; the only difference with the Plus was a few extra resident fonts.

Apple has also quietly shown a new godzilla-style laser printer to a few selected parties in hotel rooms across the country. Presumably this is a 300/600 DPI machine using a 68030, does some 20 copies per minute and comes with a built in hard disk. Up to 11 x 17 inch paper. Possibly a super secret new *Canon* JX or ZX engine. PostScript, of course.

Apple also has come out with a $1400 dot matrix printer that seems totally unusable for most serious desktop publishing. Actually, that $1400 is a lowball price. When you add in all the needed accessories, it really becomes a $2000 printer.

Ribbons are a steal at $30 each, but at this bargain basement price, you have to buy six of them at once.

The intended market for this seems to be for Fortune 500 facsimile use. It does have a color capability and can handle envelopes and labels well.

There is no truth to the rumor that a Model 28 teletype emulator will soon be available for this machine.

Meanwhile, back on the peninsula, *Adobe*, the PostScript people, are up to all sorts of neat stuff. First, they have moved a few blocks. Be sure to get their correct new address and phone number from the *Names and Numbers* section. You can still get free subscriptions to *Colophon*, some free posters on typography, and the free sets of the *PostScript Developer Guidelines* from them.

But their really big news is that Adobe has finally dropped all their insane copy protection on their line of downloadable fonts!

All fonts shipped after September are fully deprotected and will now run on any number of machines.

More important, they will not permanantly self-destruct whenever your LaserWriter password blows up.

Let's give them a full 17 credits and at least six Attaboys for this.

Now, if we can only get Adobe to drop the font lockout on *pathforall*, we would be all set. The double clip trick doesn't work anymore on V47.

Win one, lose one.

Adobe is also making a strong bid to let PostScript become the de-facto *screen* description language as well as the upcoming industry standard *page* description language. PostScript is ridiculously richer and far more powerful than either *QuickDraw* or *GemDraw* could ever hope to be. By adding a graphics chip, PostScript can become much faster as well.

There's all sorts of exciting laser printer rumors this month. A method apparently exists to modulate the dot size on those Canon laser printers, letting you dramatically improve the quality of all your photographs and most other halftones.

Some new third party accelerator boards may shortly be shipping for the LaserWriter that will dramatically speed up PostScript processing.

Facsimile machines are also about

1. If you forget your password or if flakey software trashes it, you are out $1000 and lots of time. Eased but not eliminated in v47.
2. Program errors can trip a framedevice blowup that can do subtle to bizarre things to later programs. Eliminated in v47.
3. Copypage will trash subsequent data, making copypage totally useless. Eliminated in v47.
4. Arcto blows up with too wide a linewidth or wildly wrong data; also is position sensitive with a zero radius. Not yet corrected.
5. Stuck-in-the-snow wheel spinning caused by the prefeed anticipator on large grids can simulate a major paper path failure, besides stressing the mechanism. Reduced, but not eliminated in v47.
6. Diablo emulation has nasty page creep. This is eliminated on v47. Other more subtle Diablo problems remain.
7. Multiple copies cannot be aborted on a toner or paper problem. Fixed for tray feeding on v47. A lesser manual feed problem remains.
8. An obscure "seam" problem was cured by slightly shifting the halftone masks on v47. Some high quality gray grids created on Version 38 may not image right on v47. Call me for the fix.
9. The stringwidth command took a ludicrously long 12 milliseconds to execute. Shortened to a still inexcusable 2 milliseconds in v47.
10. The font protection in pathforall could easily be defeated by a double clip. Unfortunately and stupidly, this was "corrected" in v47.
11. Slight glitches are present in some enormous Palatino font characters. This is not yet fixed in v47.
12. The miter machinery operates in a slightly different manner in v47. Some of the v38 code may not miter in exactly the same way.

Fig. 1 – Bugs in LaserWriter ROM Version 38.

to be shot out of the saddle with a new *dual-mode* machine that is able to send either PostScript or standard fax. The new machine might handle 2540 DPI, compared to the crude 216 DPI of the fax standard.

Stay tuned for more details.

Turning to shoptalk, I have moved my *Hardware Hacker* column over to *Radio Electronics* magazine, and do hope to see you over there. Emphasis there is on the traditional electronic circuits and hardware.

A reminder that we have this great PostScript BBS going great guns at (409) 244-4704. There are now several hundred free downloads available, plus a bunch of new services. Some are free; while many of the others are real bargains. Sponsors are more than welcome.

Let us start off with some real fundamental stuff for any and all of you newcomers . . .

What is a Patch?

A *patch* is some alteration that somebody has decided to make in a computer program for one reason or another. The patch might cure an obvious and major problem, or it can add a new feature, or it may simply do something that is philosophically different from the original author's personal intensions.

Patches can be both very good and very bad. Any patch, no matter how carefully done or how extensively thought out, can destroy the integrity of a program. And multiple patches can easily interfere with each other in wierd and infuriatingly subtle ways.

As practically all of the better selling programs available today run in machine language, the patches also most often have to be made as machine language changes in the final object code used at run time.

Rule number one in making any program patch is to patch ONLY a newly made backup copy of your program. NEVER patch the original factory disk!

Rule number two is to make sure the patch is intended for the particular version of the code you are currently using.

Rule number three is to verify the previous code before you install the patch. This is most often done by reading the first few bytes of your target code in the patch area.

Rule number four is to make sure that any patches you made were in fact made the way you intended and in the correct place. Do this by verifying the patch after it is installed.

WARNING — If you try ANY shortcuts, you may end up totally trashing your SCSI hard disk!

1. Save the exact IIgs machine state. Get into the emulation mode with the emulation flag set to 1, a data bank register of 0 and a direct page register of 0.
2. Verify that a Smartport controller is in the selected slot by checking for Cn01– $20, Cn03– $00, Cn05– $03, and Cn07– $00. Here "n" is the slot number. Abort the eject if no smart port controller card is present.
3. Calculate the Smartport controller entry address by adding the offset value in CnFF to $Cn03.
4. Make a Smartport STATUS $00 call with STATCODE $03 for the correct drive unit number to return the DIB Device Information Block. Read the returned Device Type Byte and the Device Subtype Byte and verify Type $01, Subtype $00 for a Unidisk 3.5 or Type $01, and Subtype $C0 for an AppleDisk 3.5. Abort the eject if you do not have an Apple 3.5 drive with removable media.
5. Issue a Smartport CONTROL command $04 with CONTROL CODE $04 to eject the disk.
6. Restore the original machine state.

For more information, see the Apple IIgs Firmware Reference Manual. Once again, do NOT skip any steps or hard disks will be trashed!

Fig. 2 – How to eject a 3.5 inch disk.

And, here is an important warning that most beginning patchees often miss – if any portion of any patch is wrong or mislocated, you MUST reset to zero and start all over again with a fresh backup copy.

Rule number five is to test your patches ONLY with scrap or expendable files, until you are absolutely sure your code is well behaved.

From time to time I have published patches for all of the various *AppleWriter* programs, as has Don Thompson in his nicely done *AppleWriter* enhancement kits.

Now this may sound obvious, but you cannot normally make a patch to a program from within that program itself. Instead, you can boot up your system master disk and get into BASIC under DOS 3.3, or else select BASIC. SYSTEM from ProDOS.

Then, you get into the monitor by doing a CALL -151. You are in the monitor when there is an asterisk in the lower left hand screen corner.

To verify the current contents of an address in machine language, you enter that address, followed by the carriage return. For instance, if you type a <1BF4 return>, the machine should respond with 1BF4– A9, or whatever other hex byte was in this particular location. Should you hit another carriage return, successive hex bytes appear on the screen.

But, note that you can only have one hex byte for each address. If eight bytes are presented in a row on the screen, these will correspond to eight *sequential* addresses in a row.

Ferinstance, a 4F98- A9 03 CF F4 29 51 8D 44 means that you have a hex $51 in hex address 4F9D.

To change a single byte, you type the address, followed by a colon, a space, the new hex data value, and a carriage return. As an example, a 2D34: EA <return> will put the hex value $EA in address location $2D34.

You can also enter multiple or sequential values at one time, up to several dozen or more, by doing a 2D34: EA 12 34 FC F5 77 . . .

The quickest and best summary of dealing with machine language does appear in chapter seven of the old user manual for the Apple II+. This topic is also covered in the IIe and IIc technical manuals and in the IIgs

Firmware Reference Manual. These are available from A.P.D.A.

Or, for more background, check into my *Micro Cookbook*, volumes I and II, or my *Apple Assembly Cookbook*. For more specific AppleWriter patches, check the ATG reprints or my *AppleWriter Cookbook*.

What is PostScript?

I can not believe the number of phone calls I got over my *Show and Tell* stuff starting on page 399 of the September CS, asking some really fundamental questions on *PostScript*. I thought I had long ago beat all these questions to death right here in this ATG column.

So, all of you regular ATG junkies, please bear with me one more time while we go over the obvious...

PostScript is a general purpose computer language from the *Adobe Systems* people that excels at doing combined graphics and text page descriptions for laser printers and high resolution phototypesetters. You normally do not have to go out and buy PostScript. Instead it is built into and included free with such laser printers as the Apple LaserWriter Plus, the QMS PC-800, and the new high end AST machines. PostScript is also available at extra cost on the *Allied Linotron* version 100 and 300 phototypesetters.

But note particularly that none of the current *Hewlett Packard* LaserJets can speak PostScript until after a quite expensive third-party lid is attached to it. Now, H-P may say they aggresively support PostScript, but their current crop of laser printers most certainly do not.

You *must* have a PostScript speaking printer to be able to make use of PostScript! As far as I know, nobody has yet written a good PostScript interpreter for a daisywheel or a dot matrix printer, although it certainly would be possible.

While PostScript has been placed in the public domain, the actual implementation for any particular printer definitely is not. Thus, while you are free to write all your own PostScript code clone, the interpreter source code for any existing machine is highly proprietary.

To a manufacturer, the only real difference between a PostScript and a non-PostScript printer is a tad extra memory and a small license fee. In this day and age, it is absolutely inexcusable to not provide PostScript on *any* stock laser printer, regardless of the list price.

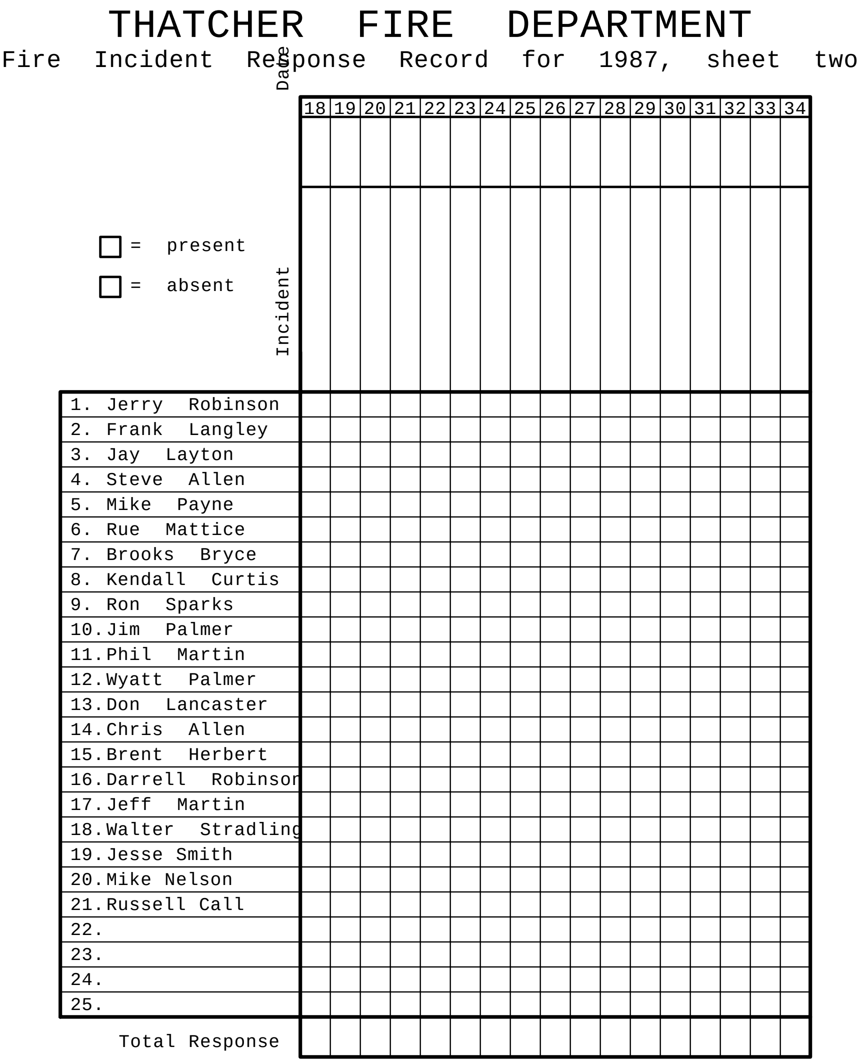
After having worked extensively with PostScript for several years now, I firmly believe that there is at least a 50:1 performance advantage of any PostScript speaking printer over anything else available.

Thus, for a non-PostScript printer to offer the same value as the Apple LaserWriter with its current $3525 street price, you would have to be able to buy that printer for $70.50.

Important advantages of PostScript include its ability to handle entire page scaleable, translateable, and rotatable mixed text and graphics in any combination, providing you with thousands of different fonts that are instantly scaleable in all sizes from 3 points to 75,000 points.

PostScript includes a sophisticated way of drawing smooth and continuous curves. These are called *cubic splines*, and are much more powerful than the usual smoothing algorithms.

And, most important of all, PostScript is largely *device independent*. Device independence means you can use any computer from a ZX80 to a



THATCHER FIRE DEPARTMENT

Fire Incident Response Record for 1987, sheet two

□ = present

□ = absent

Date	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Incident																	
1. Jerry Robinson																	
2. Frank Langley																	
3. Jay Layton																	
4. Steve Allen																	
5. Mike Payne																	
6. Rue Mattice																	
7. Brooks Bryce																	
8. Kendall Curtis																	
9. Ron Sparks																	
10.Jim Palmer																	
11.Phil Martin																	
12.Wyatt Palmer																	
13.Don Lancaster																	
14.Chris Allen																	
15.Brent Herbert																	
16.Darrell Robinson																	
17.Jeff Martin																	
18.Walter Stradling																	
19.Jesse Smith																	
20.Mike Nelson																	
21.Russell Call																	
22.																	
23.																	
24.																	
25.																	
Total Response																	

Fig. 3 – A typical rubbergrid form.

Cray-1 as the host. More importantly, the very same file that drives a 300 DPI laser printer can be *instantly* upgraded to 2540 DPI phototypesetting, simply by moving one cable.

You most assuredly do not need either a Macintosh or the Appletalk network to speak PostScript. In fact, Appletalk just costs you money, will severely limit your choice of hardware, and is often very slow.

PostScript is easily run on any Apple, Mac, Atari, IBM, or Commodore machine by using standard serial communication. Often, all you will need is your favorite word processor or comm program. These speak PostScript beautifully.

There are two major ways of using PostScript. You might either work directly with the language yourself, or else go to a canned applications program. Working with the language yourself does require some front end learning effort and does demand that you look at the printed page to see the final image.

But, the results you'll get will be unbeatably superb.

Those canned applications can be easier to learn and do give you a crude and typically very innacurate screen approximation to the printed page. On the other hand, these programs are all quite expensive. A page making program at best can only handle less than 15 percent of the total possible PostScript uses. Worse yet, you have to do everything "their way" rather than the way you really want to.

I personally feel that I can get consistently higher quality, infinitely more flexibility, faster printing, and ridiculously more control by working directly in the PostScript language. I do so using ProDOS AppleWriter 2.0 or 2.1 on a IIe or a IIgs for all of my text and *all* of my graphics, including schematics, isometrics, perspective, and even my printed circuits.

To prove my point, just tell me which canned applications program you are using, and I will send you something great looking and very easy to do in PostScript that you can not directly *create* with your program. A free book if you prove me wrong. Fair enough?

```
% 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.

/setgrid {gsave /size exch def translate size dup scale} 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

/showgrid {gsave /#vlines exch def /#hlines exch def 100 45 {eq {1}{0}
ifelse} 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 133 40
{dup mul exch dup mul add 1.0 exch sub} setscreen} def

/line1  {.06 setlinewidth} def
/line2  {.12 setlinewidth} def
/line3  {.18 setlinewidth} def

/m {moveto} def
/x {rlineto currentpoint stroke moveto} def

/r {0 x} def
/l {neg 0 x} def
/u {0 exch x} def
/d {0 exch neg x} def

/xrpt {gsave aload pop /trips exch def /dist exch def /rproc exch def
trips { gsave rproc grestore dist 0 translate } repeat grestore} def

/yrpt {gsave aload pop /trips exch def /dist exch def /rproc exch def
trips { gsave rproc grestore 0 dist translate } repeat grestore} def
```

Fig. 4 – Some sample PostScript rubbergrid utilities.

The usual way to get started in PostScript is with Adobe's *PostScript Cookbook* and my new *Introduction to PostScript* video, plus the reprints to this ATG column. From there, you can rent some hands-on time at your local school or quick-copy center.

Any IIgs Rumors?

All sorts of them. Some facts even. Apple should have sent you a letter by now telling you to go to your dealer and pick up the new ROM and the new video graphics controller chip for your IIgs. The VGC chip is only needed if you have a pink stripe in your IIgs display. While you can replace the ROM yourself, a special tool is needed for the VGC chip.

The new system ROM demands the revised System 2.0 software disk. There are no reports so far of incompatibility problems, but, naturally, they are certain to occur. Just to be sure, *save your old ROM*. Do not let your dealer steal it.

Please keep me posted on any compatibility problems.

Meanwhile, there are some really bizarre IIgs bugs newly surfacing. By now, nearly everybody knows that the "it won't print" bug can often be temporarily cured by using a *Super Serial Card* and making sure you select it on the control panel. Later on, patches can be made to your software to let you use the internal IIgs serial code.

There is also a glaringly stupid omission over on the IIgs memory expansion connector that prevents you from using DMA on most any third party memory card that has more than 1 Meg of 256K RAM chips or 4 Megs of 1 Meg RAM chips.

There is no obvious solution to this problem, so you can expect really bizarre hassles with any larger third party expansion card.

Meanwhile, there's a really cute bug in the System 2.0 software disk. It seems that if you use the launcher (otherwise known as the *luncher*, as you may want to get a sandwich while it is playing its games), and pick a ProDOS 8 application, and then return to ProDOS 16, not all of ProDOS 16 will get properly reloaded.

If your application does trash a certain obscure memory area, then certain obscure ProDOS 16 features are not any longer available. Calling them will bomb the machine. Needless to say, *Appleworks* trashes these locations. Ho Hum.

Now for that really funny bug. It seems that the *Smartport* command to eject a 3.5 inch disk is exactly the same as the SCSI command to chew up the media and spit out the seeds.

A lot of programmers have been doing quick and dirty eject routines and – surprise – will end up totally trashing their SCSI hard disks.

A summary of a safe and correct way to eject a 3.5 inch disk appears in figure two. *It is absolutely essential that you make sure that a* 3.5 *inch drive is in fact in use before you try and eject the disk*.

Many thanks to Tom Vier for his comments on this.

How can I Cut a Sheet of Paper?

It is apparently utterly and totally impossible. I had hoped to be able to give you a rundown on the clamping paper cutters this month.

Clamping cutters are needed while desktop publishing to professionally trim books or booklets and to accurately cut thick stacks of paper.

But the salespeople in the paper cutter industry are totally unreal. One screamed at me over the phone "You can’t afford one!" and hung up on me. Another one, *five times in a row*, sent me the same poorly cut sheet of paper that says they sell paper cutters, but absolutely refused to send me pricing or data on any of them. A dozen more never returned calls.

Naturally, the manufacturers tell me to "see my dealer". "My dealer" is usually 1500 miles away and does not answer his phone. I do not think he even knows how.

Or else his mother won’t let him.

Now, here’s what we need for the new desktop publishing revolution: A manual, fourteen inch, self-clamping paper cutter with a backstop that can handle half an inch of paper at a time. Is that too much to ask? I can even see paying several hundred dollars for this, but I most certainly can *not* see paying $800, particularly when a totally worthless epsilon minus of a salesperson is going to rip off $320 of that. And especially since that $800 pricing was purposely made just predatory enough to try and step you up to a motorized unit.

Well, there’s one fairly low priced clamping cutter that I have found. This is the *Dahle* model 115. I got one at our local *Price Club* for under $40. But there are two major problems with this turkey. First, there is zero mechanical advantage to the clamp. You have to push down on it, and what you push is what you get in the way of holding power.

Second, the Dahle people proudly proclaim that their cutter "never will need sharpening". What they really mean is that their cutter is *categorically impossible* to sharpen because the blade is welded in place! After a few months of use, the two blades end up chewing great gaping holes in each other. The only cure I have found for this is to add a new backstop that is three inches down from the original one.

Actually, what you really want is a guilotine style cutter with a cam-type clamp that is good for a ton or more of holding pressure.

So, for this month’s contest, either tell me about a reasonable source for clamping paper cutters or else find me a paper cutter salesperson that is not suffering from an acute case of recto-cranial inversion.

What is this Month’s PostScript Utility?

A sampling of some of my rubbergrid utilities, that are particularly handy for forms. We did talk about the rubbergrid a few issues back, so what we’ll focus on here is some simple support utilities. Figure three shows you a typical form you might like to build up, while figure four gives you several rubbergrid utilities that will greatly simplify creating the form. Finally, figure five gives you the actual PostScript code to print the non-text portions of figure three.

Note how short, quick, and easy the code is. Those *-xrpt-* and *-yrpt-* commands are incredibly flexible and extremely powerful.

The keys to the rubbergrid utilities are to use simple commands to draw lines, and to include two powerful repeat routines that will create any selected number of lines of most any length and spacing. Once on the grid, your images can be scaled to any size you like.

The rubbergrid itself is most often turned on while you are creating your form or whatever and then turned off for your final image.

One trick that beginners to rubbergridding will often miss – your font sizes and linewidths are usually *much* smaller than before. It is often a good idea to make the basic line spacing on a form "one" unit high. A suitable font to fit in a one unit high form line would have a size of 0.75 points.

Any additional rubbergrid utilities and icons are easily added for such applications as invoices, order pads, electronic schematics, ad layouts, organizational and flow charts, perspective sketches, printed circuit layouts, and isometric drawings.

Give me a call if you need more info on any of these.

```
% fire response grid

% . . . . . . . . . . . .


% requires rubbergrid utilities of figure four


90 90 13 setgrid

41 50 showgrid        % delete to turn off grid


line1

[{2 3 m 37.5 r} 1.25 25] yrpt [{14 1 m 48 u} 1.5 18] xrpt
```

Fig. 5 – Rubbergrid form code for figure three.

```
line3

2 3 m 31.25 u 12 r 14.75 u 25.5 r 48 d 25.5 l 2 u 12 l

14 3 m 33.25 u 14 34.25 m 25.5 r 14 3 m 25.5 r
```

Don Lancaster's

ASK THE GURU

December, 1987

Postscript dipdraw
Reading Apple keys
Black -vs- white write
Printing onto anything
An AppleTalk schematic

This signal indicates that the referee has just swallowed his whistle. *Apple* recently has been doing everything from 100 yard punt returns to giving all of their opposition automatic two point safeties.

Their new *Hypercard* for the Mac is really something. Then again, just maybe it is something else. Trouble is that nobody inside or outside the company has the slightest idea what the product is or what it can really do. Meanwhile, there is a rather ugly and major compatibility bug between the *Hypercard* and the *Multifinder*.

All of the unrealized potential here seems to be on the awesome side of horendous, though.

One thing that really galls me is that Apple has been witholding the *Send-PS* routine for the IIgs, while at the same time being downright rude to people using their LaserWriters on a PC or an *Atari*, let alone on a IIe or IIgs. Apparently, they are still laboring under the grave delusion that a Macintosh that's using AppleTalk is in some manner either useful or even desirable when you're doing desktop publishing.

In the real world, the Macintosh and AppleTalk route to laser printing is both slow and expensive. There can be very compelling advantages to using other host machines making a direct serial LaserWriter connection.

Particularly if you already happen to own the host. Or you simply feel strongly about using another brand of personal computer.

Reports are also streaming in that some older software will not run on the new ROM chips for the IIgs. I'll keep you posted once we get a more complete list. *Hacker II* and *Bard's Tale* and are two problem programs.

On the positive side, Apple does now have a new and a free *Vertical Markets Business Directory* out. This one lists all the software of interest to weavers, cotton farmers, for church congregations, video rental stores, law offices, etc. While a tad heavy on the real estate and accounting entries, this is a "must have" book.

The transformer is wound on a Siemens B65651-K000-R030 with a B65652 triple bobbin and a B65653 retaining clip. There are two primary windings of 35 turns each, and a single secondary winding of 70 turns. All windings are #32 wire.

The transformer must have a magnetizing inductance of 20 millihenries minimum. Leakage inductance is 15 microhenries maximum.

The 100 ohm terminating resistor is switched into the circuit if either of the AppleTalk jacks are unconnected.

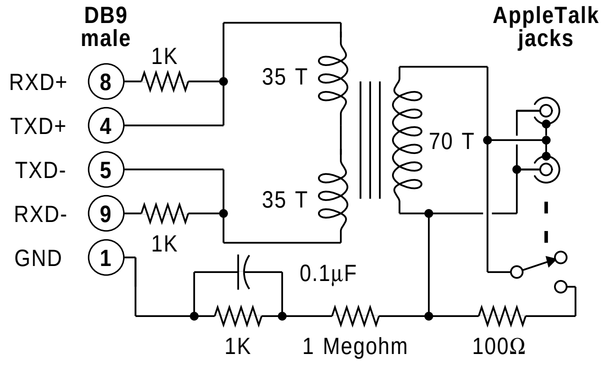


Fig. 1 – Schematic of a stock AppleTalk connector.

They also have a new *Beginner's Luck* video with an optional $7 price tag. There are coupons for this tape in many major magazines.

There's a third party outfit named *MENU* that has a pair of fine free directories called the *Apple II Guide* and the *MAC Guide*. Their intent is to direct mail sell you any and all of the listed software.

While quite useful, there are two grevious flaws here – they do not list the names, addresses, or helplines of all the original software houses, and I suspect they list only what they feel like stocking.

I'm getting these wierd calls lately over people who are afraid to use an A-B switching box on a LaserWriter, so they can run faster and cheaper than AppleTalk and still use several host computers.

The problem here is rather bizarre. Certain models of the *Hewlett Packard* LaserJet seem to have a highly flakey serial interface circuit that blows up and self destructs if you so much as think about looking at it sideways. Rather than admit they do have a very bad problem, H-P has simply blamed it all on people using A-B switching boxes.

To the best of my knowledge and belief, there *never* has been *any* time that *anyone* using an A-B box has hurt an *Apple* LaserWriter in *any* way. Dealers and salesmen may lie to you on this, because they want you to buy a Mac with AppleTalk instead.

Several readers asked if they were welcome to come visit me or even quest a *tinaja*. Wail, shore nuff.

Except that I need a week or two's advance notice. I also live a totally ridiculous distance from anywhere. Thatcher, Arizona is a sixpack away from Tucson and even further from Phoenix. (seven hours round trip by stagecoach, assuming you don't get eaten by a Gila Monster.) The local Upper Sonoran lifezone and my off-the-wall attitude both tend to be a tad of a cultural shock for easterners.

But, hail yaess. Come on an visit fer a spell.

Yes, I do speak to user groups. But on your nickel, and not on mine. My *Introduction to Postscript* video is far cheaper and makes for an interesting show all by itself.

Which is as good a way as any to sneak us into the usual advetorials. The bound sets of my *Ask the Guru* reprints remain available. Also, to find out how I make money, so is my classic *Incredible Secret Money Machine* how-to book.

Plus the usual reminders that my hardware column is found over in *Radio Electronics* (be sure and catch all the hacker superconductor stuff!), and that we have this great Postscript BBS going at (409) 244-4704.

Onward and upward . . .

What is so Special About An AppleTalk Cable?

I have sort of been wondering that myself. Inside that funny little box is a fifty cent transformer, a penny resistor and a few cents worth of wire. And little more.

Figure one shows you the schematic an AppleTalk cable drop.

At either end, AppleTalk will plug into a standard RS422 serial connector. This can be either an older DB9 connector or else a newer mini-DIN 8. All that goes over the interface is a train of pulses at a base frequency of 230.4 kilobaud. The pulse period is 4.34 microseconds.

As we've seen in previous issues, your basic "bare metal" AppleTalk communication rate is, at least in theory, around 25 times faster than using 9600 baud.

But, much of the AppleTalk software is so slow and so cumbersome that it sometimes may gobble up all of this potential 25:1 speed advantage and then some.

A digital one is defined as one transition per each 4.34 microseconds, while a digital zero is defined as two changes in 4.34 microseconds.

Each one of all the RS422 inputs and outputs of all of the parties on the line are literally tied together all of the time. Only one talker is allowed to be active at any given time, as determined by the AppleTalk firmware and software at each machine.

The circuit is fundamentally a 1:1 isolation transformer which has an operating impedance around 37.5 ohms. The 37.5 ohms comes about since you are often driving a *pair* of 75 ohm cables in parallel.

A 100 ohm terminating resistor will get automatically switched in if this node is at the *end* of the cable. 100 Ohms is used instead of 75 since it is "close enough", but will load the drivers a lot less.

The transformer shield is connected to ground by way of an R-C high frequency bypass network so that hum and low frequency noise cannot common-mode modulate the shield.

The specs for the genuine transformer are also shown in figure one, but I suspect most any old small 1:1 matching transformer may work o.k.

We will see some details on some sneaky AppleTalk substitutes in next month's column.

How do I Read the Apple Keys from BASIC?

I was suprised to get this helpline call, because just about "everybody" knows the answer to this one. It also appears in just about any Apple book, particularly in the *IIe Technical Reference Manual* and the *IIc Technical Reference Manual*. These are both available through APDA or at better bookstores.

But, here we go one more time – The *open-apple* key is the very same as game paddle or joystick button #1. The *closed-apple* key is the same as game paddle or joystick button #2.

To read the open-apple key, you do a OAKEY = PEEK (-16287).

To read the closed-apple key, you might instead perform a CAKEY = PEEK (-16286). If either result ends up greater than 128, that key is down.

The equivalent machine language locations are hexadecimal $C061 for the open-apple key and $C062 for the closed-apple key.

How can I Print Directly Onto Fabric?

You would do this the same way you would print on a box, on the side of a wall, or onto your ex-girlfriend. Naturally, you can start out by using *Applewriter* on a IIe or IIgs driving a *LaserWriter* to create your original 1:1 art. Then what?

I have found yet another astonishingly magical material that has been scunging away for many years on the back shelf of some rather obscure distributors. The whole product line is called *Merigraph*, and it is made in bulk by *Hercules* and is stocked in small quantities by *R.A. Stewart*.

The basic product is an ultra-violet curing liquid resin. It is ideal for making all your own printing plates and rubber stamps. With the LaserWriter, of course, you are no longer limited to text-only stamps, nor is there any reasonable upper size limit.

What you do is pour out a thin layer of this glop and then expose it to ultra-violet light through a transparency negative that has previously been printed on your LaserWriter.

While some special exposure light boxes are normally used, you might instead use a contact printer and a few minutes of strong sunlight.

The ultra violet light will harden the front of the sheet where the text or images are to appear, down to a reasonable depth. Next, you flip the whole works over and then expose the entire back to harden it. You end

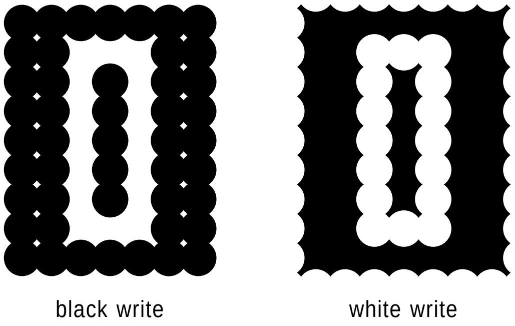


Fig. 2 – Differences in laser printing engines.

up with a giant rubber stamp that has a flat, rubbery back, and super sharp characters and images on it.

Finally, of course, you can mount the resin on a backing and a handle. You then use it like any other rubber stamp, selecting your ink chemistry and color to suit whatever it is you are printing on.

A similar material is also being used for the hot new *Santa Claus* machines that will give you nearly instant three dimensional CAD model prototypes directly off the disk of your Apple or other workstation.

Needless to say, keep me informed on what you are up to with this fantastic glop. The potential here would seem to exceed the *Omnicolor* and *Kroy Kolor* materials that we looked at in earlier columns.

I am firmly convinced there are all kinds of other "neat stuff" chemicals, materials, and processes that are now lurking "out there" just waiting for one of you *Computer Shopper* readers to discover and suitably misapply them in unintended ways.

So, for this month's contest, just tell me about anything at all in the way of obscure neat stuff. Something that a veterinarian, a sign painter, a blacksmith, a weaver, a stained glass artisan, or a potter might use every day, but that is totally unknown elsewhere in the real world.

As per usual, an *Incredible Secret Money Machine* to the best twenty entries, and an all expense paid (FOB Thatcher, AZ) *tinaja quest* to the best entry of all. We will share all of the results with you in later columns.

What is the "Black Write" Versus "White Write" Laser Printing Controversy?

There sure seems to be bunches of misinformation on this topic, not to mention all of the outright lies. Let us try to set the record straight.

It turns out there are two different methods of designing a laser printer engine. In a *black write* engine such as the *Canon* CX or SX, any spot the laser beam hits on the drum becomes a *black* spot on the paper. In a *white write* engine such as the *Riccoh*, any spot the beam hits becomes a *white* spot on the paper. So far so good.

The problem comes in when you try to put a round laser light spot into a square hole on the paper. The spot usually has to be somewhat larger than the square. But if it gets too large, very ugly things will happen.

Figure two shows you what will typically occur. In the case of the black write machines, it is difficult to get very solid blacks over the larger fill areas. This happens because some corner cutting would seem inevitable. But this difficulty gets eased bunches by selecting the correct paper and by using a toner cartridge that is at its maximum blackness *after* its second refill with a good third party product.

Contrary to some tales that are circulating, solid black fills can most definitely be obtained with a black write engine. I do it all the time. It does take some care and attention to detail to accomplish this, though.

Now, in the case of a white write machine, all of the black fills are totally solid most of the time. The problem here is that any small typography, particularly anything less than seven point, will totally wash out and become nearly illegible.

You'll also find a tendency to white out the lightest shades of gray with a white write system.

There are two other big differences between the black write and white write engines. A black write laser engine will normally have a much *longer* laser diode life, since most parts of most documents are white rather than black.

A second problem common to both methods, is that linear gray shading is not possible, particularly with the denser screens. The black write systems tend towards *darker* grays than you would expect, while those white write systems produce much *lighter* grays than you would expect.

Canon seems to have eased the black write hassles with its second generation SX engine as used in the new LaserJets. This most likely was accomplished through a squarer spot, special modulation techniques, and a high energy toner formula.

The bottom line here? My personal opinion is that most of the existing black write systems are far and away the better choice for most users most of the time. As the improved engines become available that use squarer spots, the differences between the two should eventually vanish.

What is this Month's Postscript Utility?

Hmmmm. Stan Veit does keep on threatening to throw several of his alligators into one of my tinajas if I don't start getting this column out on time. I have got bunches of not-quite-prime-time Postscript stuff just sitting here in the wings.

Stuff like a full perspective (that even includes the lettering!), printed circuit layouts, compiling tricks that dramatically speed up Postscript, and a brand new set of utilities for most any host computer.

The trouble is they are all "pretty nigh but not plumb". Let me know if you are personally interested in any zeta testing.

Instead, this month we'll go back to one of my old favorites. This is my

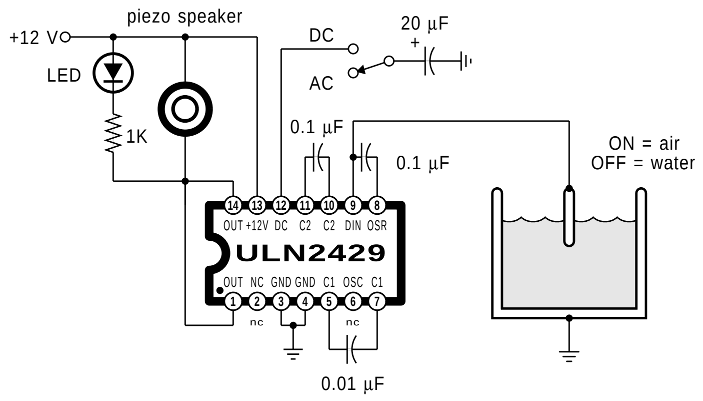


Fig. 3 – A typical use for my dipdraw routine.

automatic DIP pictorial drawing routine. Figure three shows you a typical integrated circuit, as is used in an electronic schematic. The *dipdraw* routine accepts two position values, a size numeric, a title string, a top callouts string, and a bottom callouts string. Dipdraw will then automatically draw the correct dip of the correct size in your chosen location.

The *Postscript* dipdraw coding is shown you in figure four. A zillion previous examples appear in all of my *Hardware Hacker* columns that appear in *Radio Electronics*.

Complement bars on the callouts are all automatically handled with a leading slash. A maximum of three characters per callout is typical. Four can sometimes be used if the adjacent callouts are thin enough.

If the integrated circuit has 36 or more pins, it is drawn one unit higher than are all the packages with fewer pins. This preserves a good shape factor for LSI chips.

The width of the device number font can be changed if you need to force extra characters into a smaller package outline.

The drawing is normally scaled so that one unit represents the spacing between two pins. The position is always referred to pin number one. If you are drawing many different integrated circuits, you can use a special dictionary so that calling, perhaps *dip27256*, will automatically get the correct values and drop them in place with only a few keystrokes.

The dipdraw pictorial is opaque. Normally, you will place your wiring *early* in your textfile and your dipdraw routines *later*. This way, the dip pictorial will overlay the wiring, and each wire will exactly stop at its own respective pin circle.

Special overlays might later be added, such as individual inverter, gate, or op-amp symbols. The device number must, of course, be moved outside the package if you do this. These can also be dictionary based.

Note that the final print quality and the convenience of this routine will dramatically exceed what you can usually do with any screen oriented drawing program, and that you can use this on any make and any model of personal computer.

Write or call for more info.

```
% dipdraw - draws a dip integrated circuit.

% Copyright c 1987 by Don Lancaster & Synergetics, Box 809, Thatcher
% AZ, 85552, (602) 428-4073. All rights reserved. Personal and non-
% commercial use permitted so long as this header remains intact.

% Enter with currentpoint set to pin 1 and scale set so that 1.0 = distance
% between pins. Then do a numpins-(name)-(hipins)-(lopins) dipdraw. Pin
% callouts preceeded by a "/" will get complemented.

% main dipdraw entry:

/dipdraw { save /snap exch def /hipins exch def /lopins exch def /chipname
exch def /numpins exch def /howlong {numpins 2 div cvi 1 add} def /howhigh
{4 numpins 36 ge {1 add} if} def /stub {howhigh 1.4 sub 2 div} def

% internal service subs start here:

/pinproc {numpins 2 div cvi{newpath 0 cpos 0.37 0 360 arc gsave 1 setgray
fill grestore 0.067 setlinewidth stroke  pin# 5 string cvs dup stringwidth
pop 2 div neg cpos 0.2 sub moveto show 1 0 translate /pin# pin# dir add def}
repeat } def

/stretchprint { dup stringwidth pop 2 div neg exch length 1 sub stretch mul
2 div sub 0 moveto callout (/) anchorsearch true eq {currentpoint exch
stretch add exch moveto pop dup /callout exch def stringwidth pop callout
length 1 sub stretch mul add /barwide exch def 0.033 setlinewidth gsave
currentpoint 0.55 add moveto barwide 0 rlineto stroke grestore} if stretch
0 callout ashow} def

/pincallouts{0 vpos translate {workstring ( ) search true eq {/callout exch
def pop /workstring exch def callout stretchprint 1 0 translate}{dup /callout
exch def stretchprint exit } ifelse} loop} def

% actual dipdraw process starts here:

% . . . . the outline:
gsave  1 setlinecap 1 setlinejoin currentpoint translate newpath -.55 .45
0.15 0 360 arc fill newpath -1 howhigh 2 div 0.7 -90 90 arc 0 stub rlineto
howlong 0 rlineto 0 howhigh neg rlineto howlong neg 0 rlineto closepath
0.36 setlinewidth stroke

% . . . . pin circles and numbers:
/Helvetica-Bold findfont [0.4 0 0 0.55 0 0] makefont setfont gsave /pin# 1
def  /dir 1 def /cpos 0 def pinproc grestore gsave /pin# numpins def /dir
-1 def /cpos howhigh def pinproc grestore

% . . . . pin callouts:
/Helvetica findfont [0.35 0 0 0.6 0 0] makefont setfont /stretch 0.033
def gsave /workstring hipins def /vpos 0.6 def pincallouts grestore
gsave /workstring lopins def /vpos howhigh 1.05 sub def pincallouts grestore

% . . . . device number:
/Helvetica-Bold findfont [1.2 0 0 1 0 0] makefont setfont /stretch 0 def
gsave numpins 2 div 1 sub 2 div howhigh 2 div 0.33 sub translate chipname
dup /callout exch def stretchprint grestore

% . . . end cleanup:
grestore grestore clear snap restore} def


% . . . . . .

% Dipdraw demo - remove before use
150 200 translate 25 dup scale 0 0 moveto
14 (ULN 2429)
(OUT +12V DC C2 C2 DIN OSR)
(OUT NC GND GND C1 OSC C1) dipdraw   showpage
```

Fig. 4 – My Postscript dipdraw routine.

Don Lancaster's

ASK THE GURU

January, 1988

Postscript boxifier
Laser printing papers
New BASIC for the IIgs
AppleTalk cable substitutes
Monochrome HIRES graphics

Apple has just released a very powerful new BASIC that you can use on your IIgs. Yes, it does access all of the resources of the IIgs, including the full memory and all of the new toolbox. Yes, it includes a *printusing*. Yes, it allows long variable names and labeled *goto* and *gosub* calls.

No, it is not a *Microsoft* product. It seems to be a latter day "channeled" reincarnation of that *Apple Business Basic* long ago used on the Apple III. Cost is $50 and it is now available directly from *A.P.D.A*.

Seperate licensing of a runtime module is available, so that users of your custom software do not have to buy the program themselves.

Some negative points include total incompatibility with all the existing Applesloth programs, the inability to run under older ProDOS or DOS 3.3, rather slow booting, and the lack of any downward compatibility as needed to run on anything but the IIgs.

Apple has now announced version 47 LaserWriter upgrade kits. These speed you up by one-third and will eliminate some bad habits. At a suggested list of $320, they are definitely cost effective if you are using your machine two hours per day.

There is a new and free Apple user group publication that's called *Tech Tidbits*. But the issues I have seen to date are so full of errors, omissions, and outright misinformation that they could only be a secret IBM plant.

Their most heineous crimes so far lie in perpetuating the outright myths that AppleTalk is needed to drive the LaserWriter, or that up to $150 worth of custom cables are required to use AppleTalk at all.

IWEM begins with a series of comments and a revision history.

A test is made to see if the IWEM dictionary is already installed. If not and if a valid password, a persistent download is begun. Otherwise, the download is aborted.

The main emulation code is defined as /_WDJ_. When called, this begins the emulation for the rest of the job.

The IWEmdict is defined and begun. Countries are defined and a default country is selected. Twenty mouse font characters are then defined, followed by a grafFont character set.

The Courier font is then modified to print four extra characters, the combined AE in upper and lower case, regular and bold.

International character remapping code is then defined, as needed for the various languages.

The printfile routine is defined as the main job loop. This reads and prints characters until a control-D end of file.

Routines are defined for InitVariables, ResetPrinter, Reset_Tabs, initjob, initpage, hexout, and doCmd.

Character swap routines follow, as needed for each language.

Various interpreting and control definitions are next, in turn followed by formfeed, carriage return, backspace, tabs, and character width stuff.

A tab dictionary is defined to remember tab positions. This includes a sort routine to put the tabs in order.

An escape dictionary is defined to handle all of the escape sequences being emulated and their service routines.

All dictionaries are closed and the program is terminated.

Fig. 1 – A summary of the IWEM imagewriter emulator text file.

Neither of these myths are true. The LaserWriter can easily by driven by *any* serial port from *any* computer. Under certain circumstances, serial communication can take place as much as seven times *faster* than with AppleTalk. As we will see below, if you really do want to use AppleTalk, there are all sorts of free to $5 alternatives to those $150 cables.

The *Tidbits* purportedly have a list of Apple II publications, but they do omit *Open-Apple*, which is far and away the best and most important resource of them all. Both *Apple Assembly Line* and *Call A.P.P.L.E* are also incredibly omitted. So is the *InCider* magazine.

Then, they have the utter arrogance to suggest that *AppleWriter* is not useful on the IIgs, when in fact it is far and away the finest word processor available for use on the IIgs today. We have seen the simple three byte patch needed right here in our earlier ATG columns. Over the past year, this patch has beeen widely distributed, as have many other AppleWriter improvements.

Finally, the only useful Postscript resources they cover are the red and blue books. Our free PostScript BBS and free help line is not mentioned, nor are any of several others.

The *Tech Tidbits* idea does seem sound, if only they can get their act cleaned up. Time will tell.

The *Letraset* people have recently introduced some great new desktop publishing tools and materials. Unfortunately, their "sales" department was so snotty to me that I have completely forgotten what these products are or where to get them.

PostScript is rapidly going for all of the marbles. Besides its use as a standard page description language and as a newly emerging standard screen description language, look for PostScript facsimile replacements, the BBS graphics exchange formats, all the PostScript engraving and sign making machines, and even some new PostScript software you can use for printed circuit layouts.

I will throw in the usual reminders that my hardware hacker column has recently been moved over to *Radio-Electronics* magazine and that we have this great PostScript BBS going on at (409) 244-4074. Meanwhile, you can write or call for a new brochure or two that I have put together.

This month, I've cracked some but not all of the IIgs AppleTalk secrets, so let's have at it . . .

How Does AppleTalk Work on a IIgs?

There has long been a dearth of information on the AppleTalk drivers for the IIgs. Some resources I have found useful are the original *Inside AppleTalk* notebook, chapter six of a very obscure document known as the *Cortland Preliminary Notes*, and the *AppleTalk PEEK* Mac program.

You can check A.P.D.A. for all the latest IIgs and AppleTalk tech info.

The PEEK program uses a second computer to sit on the line and sniff the fumes on ther way by. It is most handy as a debugging tool and to find out what is really going on.

Three AppleTalk drivers are now supplied with release 2.0 of the IIgs system software. These three drivers are for the Imagewriter under AppleTalk, for a non-existent file server, and for the LaserWriter. The drivers are installed by using the IIgs AppleTalk *Chooser II* accessory

I guess I was most interested in the LaserWriter driver. I much prefer using straight serial connections with the LaserWriter, since this is simpler, cheaper, more flexible, and often faster than using AppleTalk. But you might want to go the AppleTalk route if a IIgs or two and a Mac or two are going to share the same printer.

In its unaltered form, this stock LaserWriter driver *forces* you to use an Imagewriter emulation mode, and *prohibits* you from sending any raw PostScript. Which is handy for rank beginners and for use with several programs that do not know anything about PostScript, but is otherwise nearly totally useless.

Fortunately, there's some simple yet sneaky tricks we might play to make the IIgs AppleTalk driver for the LaserWriter actually useful.

To use your stock driver as an emulator, you simply send your first character to $C700 and then route all succeeding characters to $C707. Just like a plain old serial port. The drivers are built in and ready to go in AppleWorks 2.0 and higher. From Applesloth, you can do a PR#7 and you are home free.

To print from *AppleWriter* in the AppleTalk Imagewriter emulation mode, install AppleTalk and use the Chooser II. Then click into BASIC. SYSTEM. Follow this with a PREFIX, D2 command and a -AW.SYSTEM.

Which will now switch you into AppleWriter without disconnecting the AppleTalk network. Finally, do a PD7 before you actually print.

It is a real bear to design your own AppleTalk driver. But it is a fairly easy process to modify portions of the existing LaserWriter driver so you can make it do anything you like. No, you do not need anything special in the way of machine language or IIgs expertise for this.

The two facts that let us get sneaky are these: (1) The initializing routine sent to the printer is all contained in an unprotected, stock and standard PostScript text file named IWEM; and (2) Each time that you begin printing a new document, a single PostScript command _WDJ_ is *always* sent out as the *first* printer instruction.

A summary of IWEM does appear in figure one. This is an absolutely straight adaption of Adobe's generic daisywheel driver, customized for the Imagewriter. That mysterious _WDJ_ initializing command is used to turn on a dictionary and start the actual emulation for each job. Very handily, the emulation does end with each file printing and returns on back to native PostScript.

So, to get sneaky, all you have to do is rearrange IWEM to suit yourself. Make a backup copy of your system disk and plainly label it as modified. Then, rename IWEM as IWEM.ORIG. Grab IWEM.ORIG with AppleWriter, fix it, and save it back as IWEM. Then label the change.

You can do anything you like here, but remember that the first command that will be sent to the LaserWriter will *always* be named _WDJ_. Any new IWEM routine absolutely *must* do a persistent download of *something* that is named _WDJ_.

Figure two does show you how to convert IWEM into a workable clone of SEND.PS. All that I have done here is renamed the old _WBJ_ proc as *imagewriter*, and then defined a new dummy _WBJ_ procedure that does nothing at all.

With your newly modified IWEM, everything sent to the LaserWriter is assumed to be PostScript and gets handled as PostScript. Should you ever actually want to get a printed listing of your PostScript code, just temporarily add the command *imageWriter* to the start of your file.

To convert the stock IWEM AppleTalk imagewriter emulator into a program that will send PostScript code to the Laserwriter . . .

1. Make a backup copy of your IIgs system master v2.0 disk. Label this disk as modified for direct PostScript use.
2. Rename APPLETALK / IWEM as APPLETALK / IWEM.ORIG.
3. Load APPLETALK / IWEM.ORIG into AppleWriter.
4. Rename the PostScript procedure "/_WDJ_" as "/imagewriter".
5. Immediately following the above routine, define an "empty" procedure as "/_WDJ_ { } def".
6. Save the file as APPLETALK / IWEM.

When the modified emulator is installed, PostScript code can be directly sent over AppleTalk via the usual PR#7 or PD7 methods.

To print the PostScript file rather than executing it, simply prefix the entire file with the "imagewriter" command.

Fig. 2 – Converting IWEM to a SEND.PS clone .

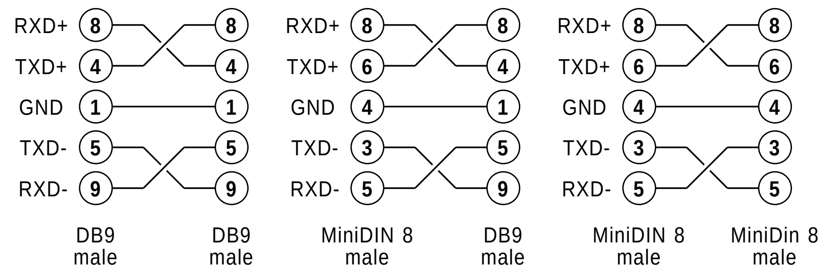


Fig. 3 – NULL modem cables for use as AppleTalk substitutes.

Instead of executing the code, it will simply get printed in the emulation mode. Sneaky, huh?

What are the Options to Stock AppleTalk Cables?

If you are going to believe Apple, cabling that lists for $150 must be bought to use AppleTalk at all, and you absolutely must use AppleTalk in order to use the LaserWriter. Both of these concepts are outright myths.

There are at least five different ways of beating the $150 AppleTalk access tab. The first and cheapest, is to not use AppleTalk at all, since it can slow you down and can very severely limit your choice of host machines and output devices.

In the case of a LaserWriter, a serial A-B box often works just fine and is far more flexible.

Your second option is to build your own cables, following the easy schematic we looked at last month. The folks over at *Redmond Cable* can supply you with the connectors.

Your third option is to substitute unused telephone lines for all the expensive cables. In the long run, the *only* successful local area networks *must* use existing telephone cables; anything else will automatically price itself out of its market.

At any rate, there's an outfit called *Farallon Computing* that will be very happy to sell you alternatives to the stock AppleTalk that use ordinary phone line or other twisted pair and that cost far less than the original.

If you only have one computer connected to one printer that both share the same grounded AC power outlet, then you do not need anything fancy at all in the way of an AppleTalk connector. All you'll really need is a stock NULL modem cable that sells for as little as $5.50 from such outfits as *Microcomputer Cable*.

Figure three does show you three different cables that differ only in their chosen connectors. So, our fourth alternate is to use a plain old NULL modem cable. But remember that this will only work as an AppleTalk substitute if there are two and only two nodes present on your network, and if both are *always* plugged into the same grounded AC outlet.

Fifth, and finally, all you really need to work with AppleTalk is some doorbell wire. Figure four shows you an AppleTalk "cheater cord" that will work with three or more drops. This can *only* be used if any and all of the computers and printers to be used are *always* plugged into the very same grounded AC outlet strip.

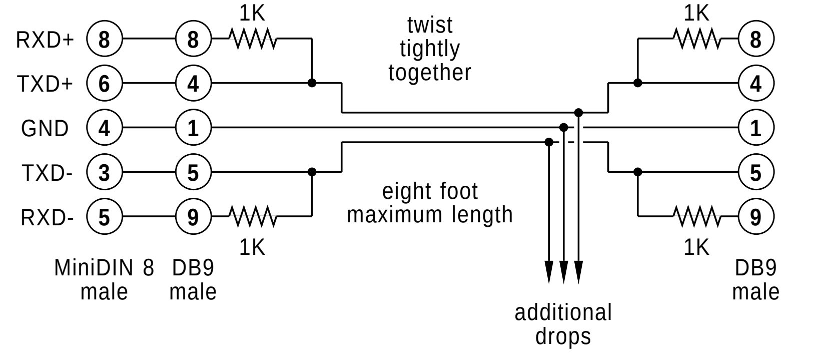


Fig. 4 – A multi-station AppleTalk "cheater cord" .

What is the New IIgs Monochrome HIRES mode?

At long last, they have figured out how to get rid of the green and purple color fringes on the HIRES screen. This beauty is in the latest release (#29) of the *Apple II Technical Notes*. The notes are available directly from Apple, from your local user group, or from many BBS systems.

All you do is select monochrome and 40 column from the control panel and get into graphics and HIRES. Then do a read or a write to hex location $C05E, otherwise known as a PEEK (49246).

To get the color back, just read or write to location $C05F, or else do a PEEK (49247).

Or, to get more technical, to get into this mode, whap soft switches $C050, $C057, $C00C, and $C05E. Then set bit 5 in $C029 by loading $C029, then ORing it immediate with #$20 and then resaving it to $C029.

There's a similar 80 column color fringeless mode you can get into by whapping the 80 column soft switch. This has to be done inside a program, rather than directly from the monitor.

The result can be a most dramatic improvement in monochrome HIRES displays. You still cannot have a full color HIRES image with black and white mixed text below, though.

Show Me An Integer Square Root Routine

I get lots of calls for fancy math routines usable from Apple machine language. Probably the best way to handle most of this today is with the SANE numerics *(Standard Apple Numeric Environment)*, which give you nearly all of the high precision math and trig functions.

You'll find SANE all ready to go in both the IIgs and the Mac tool boxes. You can also get disk-based SANE assembly language routines for older Apples from A.P.D.A. These routines are not particularly fast, but they do the job and do it well.

16-bit integer square root code. An accurate and astonishingly fast (only 38 microseconds on a IIe; 16 for the IIgs!) integer square root appeared in the September 1987 issue of the *Apple Assembly Lines*.

This jewel uses table lookup and some off-the-wall thinking to come up with a fast, elegant, and not overly long (less than 1K) solution.

In general, the algorithm for many square root routines is based on your taking the root series (1 4 9 16 25 ...), and then taking the *first difference* between each root (3 5 7 9 11 ...), and finally noting that the *second difference* is always exactly two. Reverse and iterate the process to calculate the needed root.

Tell Me All About Laser Printing Papers

Practically any uncoated or nonmeltable paper can be used with a laser printer, although a very careful selection and testing is often needed to get the best possible results.

Let's start off with Hornsnaggle's two laws of paper buying: (1) Any of the old-line printing paper supply wholesalers will eat you alive with outlandish prices, totally ridiculous minimum orders, the indecipherable pricing, oddball substitutions, and inexcusable delays.

These outfits do, however, have excellent and free sample kits that are definitely worth ripping off.

And, Hornsnaggle also tells us that (2) the *basis weight* of a paper is a totally meaningless concept invented by a consortium of paper salesmen and specifically designed to confuse and confound you. The weight of a paper has nothing whatsoever to do with how thick it is, how rigid it is, how it feels, its opacity, or even how much a ream of it will weighs.

Your first goal should be to find a cheap source of plain old Xerox copy paper, which will work just fine for everyday uses. Out here in Arizona, there are "warehouse" style grocery stores called *Price Clubs* that will sell you ten reams of excellent copy paper (Such as *Simpson* or *Westland*) for $17 a case, or $1.70 per ream.

How they can do this is a tad of a mystery, for their retail price is far below anyone else's large quantity wholesale price.

This paper is not the whitest and has a slightly greasy look about it. It also is not really opaque enough for high quality two-sided printing. I use plain old offset paper (Such as *Husky* 60# at $3.00 or so per ream) for my self-published *Ask the Guru* reprints and all of the other books we custom publish here. Offset is a thicker and whiter paper with a better feel and has negligible print-through.

I get paper from a new-age outfit called *Paper Plus*, a nationwide self-service walk-in chain that offers a wide variety of quality papers cheap and in very small quantities. I also get specialty papers from them for the laser printing of fluorescent bumperstickers, letterhead and stationary materials, the blank certificate forms, parchment stocks, invitation blanks, astrobrights, covers, and label stock.

If you know of any other chains similar to *Paper Plus*, please let me know so that we can pass this on to interested readers. As usual, there will be books and/or tinaja quests to the providers of the best info.

The best sources I have found for pressure sensitives are those *Wausau* Print-Paks. These are available both coated and uncoated, in fluorescents, whites, and in some special colors and finishes. The price is around $8 per hundred sheets.

There are also several specialty papers intended specifically for laser printing that can give you outstanding results. At, of course, outlandish prices. These prices should start to drop as these papers become more in demand and wider known.

Examples of very high quality laser printing papers include *Hammermill* Laser Plus, *Gilbert* Nu-Tech, and *ProTech* Laser Bond. Those *ProTech* people also have laser printable overhead transparency materials.

Speaking of which, self-stick clear acetate or mylar materials can usually be manually fed. Mylar is the better choice since it is more stable and melts at a higher temperature. One source is *Dick Blick*.

Do *not* attempt to feed an acetate sheet that does not have a thick paper backing, or it is certain to melt and glop up the works.

Another "paper" that you definitely do not want to run through your laser printer is that *tyvec* polypropylene paper as used for some floppy disk holders. One minor problem is that the toner falls right off it. But that's completely academic, since the heat fusion rollers will convert this paper

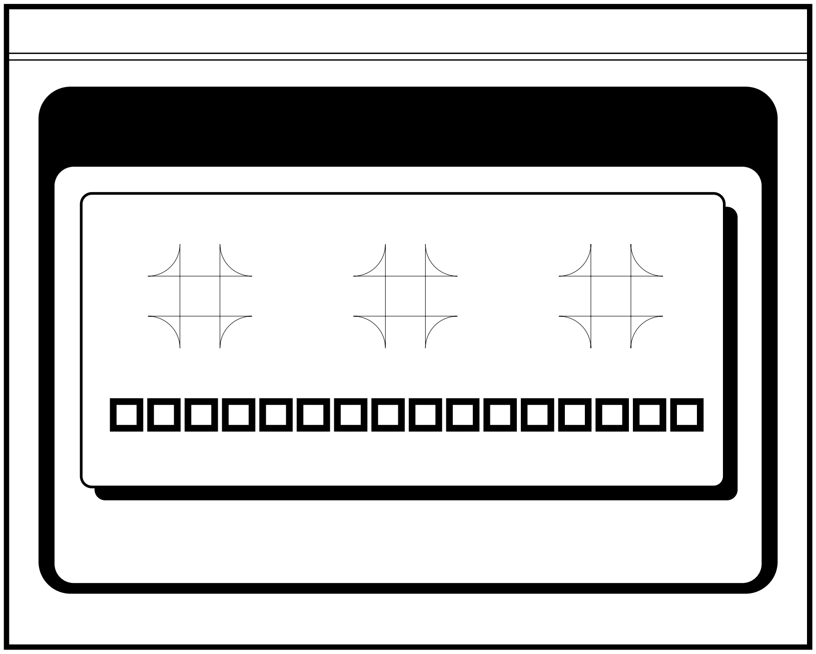


Fig. 5 – Some uses for my PostScript boxdraw stuff.

into a black lagoon style of gloppy syrup. Real yummy.

Two of the many trade journals that have lots of free paper and paper selection information in them are *Printing Impressions* and *Electronic Publishing*. As usual, see the *Names and Numbers* section for all of the needed addresses.

What is this Month's Postscript Utility?

This month, we'll pick up a very simple and useful set of my *boxdraw* routines. These will automatically put a box around anything you want. The box can have square or rounded corners of any size or weight, with or without a double hairline header.

You can also do radiused, filletted, or cusped corners for special effects.

Figure five shows you some of the sample boxifier output, while figure six gives you the PostScript code.

The -bl- (boxleft) gets measured from the left edge of the paper, while -bw- (boxwidth) is the width of the box between its left and right sides.

The -bt- (boxtop) is measured from the bottom of the page, while the -bh- (boxheight) is the height of the box between its top and bottom.

An additional -bc- (boxcenter) variable is defined for your later use. It is particularly handy for centering additional graphics or text inside of your drawn box.

```
% boxdraw box and border drawing utilities
% . . . . . . . .

%  Copyright c 1988 by Don Lancaster & Synergetics, Box 809, Thatcher
%  AZ, 85552, (602) 428-4073. All rights reserved. Personal and
%  non-commercial uses permitted so long as this header remains both
%  present and intact.

%  To use, enter -bl- -bw -bt- -bh- -brad- -blw- boxdraw.
%  For a hairline, use -hd- hairdraw after drawing the box.

%  Large positive -brad- values give you rounded boxes.
%  A -brad- value equal to -blw- / 2 gives you a radiused box.
%  A zero -brad- value gives you square corners.
%  A negative -brad- equal to -blw- / 2 gives you a corner fillet.
%  Larger negative -brad- values give you cusp style corners.

/boxpath {/blw exch def dup 0 eq {pop 0.001} if /brad exch def /bh
exch def /bt exch def /bw exch def /bl exch def /br bl bw add def /bc
bl bw 2 div add def /bb bt bh sub def /strt br bl add 2 div def newpath
strt bt moveto br bt br bb brad arcto br bb bl bb brad arcto bl bb bl
bt brad arcto bl bt strt bt brad arcto closepath blw setlinewidth} def

/boxdraw {boxpath stroke} def
/boxfill {boxpath fill} def

/hairdraw {gsave /hd exch def 0.5 setlinewidth bl bt hd sub moveto bw 0
rlineto 0 2.5 rmoveto bw neg 0 rlineto stroke} def

/bc {bl bw 2 div add} def

% //// DEMO - REMOVE BEFORE USE /////

/bl 200 def
/bw 175 def
/bt 500 def
/bh 240 def
/brad 7 def
/blw 2 def
/hd 25 def

bl bw bt bh brad blw boxdraw hd hairdraw

gsave bl 10 add bb 15 add translate

% {any stuff to be locked inside the box goes here}

grestore showpage
```

Fig. 6 – Some of my Postscript boxdraw routines.

One handy thing about the boxifier is that you can most lock any text or graphics to your box and it can then automatically track for you.

To do this, you use the -bl- (boxleft) and the -bb- (boxbottom) variables to translate the position of whatever is to go inside.

A sneaky additional trick is to predefine the box but defer drawing it until *after* all its contents are complete. You can then redefine -bb- so it will automatically use up exactly the right amount of space that all its contents may currently demand.

This gives you "perfect" text fitting without knowning ahead of time just how many lines you are going to put into the box. The bottom of the box will automatically move down with the text, provided that you include a redefinition of the -bb- (boxbottom) variable that changes with the last line text position.

On the other hand, if you do *not* need any deferred operation, you can simply input all the numeric values before calling -boxdraw-, rather than pre-defining all of your variables. This is faster and easier for plain old fixed height boxes.

The -brad- box radius variable lets you perform all sorts of very sneaky tricks. If you do use a large positive value for -brad-, then you will get a box with the usual rounded corners.

If you do use a -brad- value that is exactly one-half of the box linewidth, then you will get a radiused box with square inside corners and round outside ones. This looks best with fairly fat borders.

If you use a zero -brad- value, then you get perfectly square corners. Finally, you can use negative values for -brad-. These will give you fancy inverse cusps as shown you in the middle of figure five.

Your boxes can be repeated inside boxes. Make the outside one black and the inside one white, and you have a fancy double border, similar to the ones that I usually use on my *Synergetics* ads.

The boxifier can also be repeated for such things as fancy continuous borders. Let me know what else you can come up with on this.

Don Lancaster's

ASK THE GURU

February, 1988

New LaserWriters
Postscript point grid
Printing onto aluminum
Understanding IIgs modes
IIgs programming resources

Barring any further delays, *Apple* is about to announce a pair of fresh *PostScript* speaking laser printers. The current *LaserWriter Plus* is being upgraded to use the new *Canon* SX engine. This new engine is far and away the best choice available today for low end laser printing.

The SX has been designed from the ground up as an actual laser printer, rather than being thrown together from recycled copier parts. The new SX is smaller, more rugged, and costs less than the old CX. It holds more toner per cartridge and gives much more solid blacks, particularly over any larger fill areas. The paper path is simpler and less jam prone. The trays hold more paper. The manual feeding is far easier. It collates.

Envelopes are less of a hassle to print and come out better looking.

I have not taken one apart yet, but presumably there has been a computer board redesign to pick up more modern components, an easing of the password blowup travesty, additional memory for downloadable fonts, and modestly faster speed. The Version 47 PostScript ROM firmware is also presumably still being used.

Yes, the SX cartridges can still be refilled for under $10. But some early rumors report that the SX cartridge drums are now ridiculously easier to scratch than they were before on the older CX.

The *Apple* high end machine is still a bit of a mystery at this writing. It is presumably very pricey, with a faster and larger PostScript engine intended for shared group use, with a 68020 main processor, a hard disk provision, and lots of fonts. Plus a few other bells and whistles.

We'll see lots more details on both of these new machines after I have a chance to do some further snooping and user testing. Stay tuned.

Meanwhile, several of you helpline callers have had some very positive comments on the new NEC LC-890 PostScript printer that is intended to compete one-on-one with the LaserWriters. While I haven't yet received one for testing to date (hint... hint...), this beast apparently uses a liquid crystal shutter instead of a scanning laser, and seems to use an in-house engine design.

I'd tend to wonder what the long term reliability and pixel uniformity of the new liquid crystal shutter will be. My prediction here is that the shutter will have its own set of rather unique problems. Time will tell.

Meanwhile, watch out for the hype from others. Several reviewers have recently been unfairly comparing the other new laser printers against non-upgraded and slow old LaserWriters that still use those obsolete version 38 firmware ROMs.

So, before you believe any of all the hype and certainly before you buy any laser printer, here are some essential guidelines: (1) Make absolutely certain the machine and all its electronics has been designed from the ground up to *internally* speak PostScript as its native language; (2) Get sample printouts of *both* 6 and 600 point text; (3) Do check out the filling in of large black areas; (4) Perform tests of your actual required output on the target machine, (5) Make certain that a third-party toner refilling process is available to keep your toner costs under 0.7 cents per page; and (6) Run some of the 200+ free downloads on the machine you can get off our free PostScript BBS at (409) 244-4704.

If you do not already have access to a PostScript speaking printer, the best way to get started on all this is with Adobe's *PostScript Cookbook* and with my own *An Introduction to PostScript* video. The latter also will show you how to do cartridge refills and will pay for itself three times over on your first refill. I have both of these in stock here, along with lots of other PostScript goodies.

Oh, the utter shame and horror of it all. Apple has stupidly dropped AppleWriter from their developer price lists. Why this is so is a real mystery, since AppleWriter is still the only genuinely useful word processor that is currently available for the IIgs, besides being an outstanding desktop publishing choice.

Yes, I will admit that there are new IIgs word processors that can freely intermix totally illegible mauve and fuscia text on an orange background. But all of these new programs are so stupendously slow that they're only useful when you are writing a term paper on glaciers, snail geriatrics, or political reform.

Besides, of course, lacking those powerful self-prompting glossaries,

Apple Assembly Cookbook (Lancaster)
Apple Assembly Lines (S-C)
Apple IIgs Firmware Reference (APDA)
Apple IIgs Hardware Reference (APDA)
Apple IIgs ProDOS 16 Reference (APDA)

Apple IIgs Toolbox References I and II (APDA)
Apple Programmer's Workshop (APDA)
Apple Programmer's Workshop Reference (APDA)
EDASM Assembler Tools (APDA)
Exploring the Apple IIgs (Little)

Micro Cookbooks I and II (Lancaster)
ORCA/M Assembler (Byte Works)
Programmers Introduction to the Apple IIgs (APDA)
Programming the 65816 (Eyes and Litchey)
S-C Assembler (S-C)

Fig. 1 – Some Apple IIgs programming resources.

the full macros, and the magic of the WPL executive supervisory language. No way can they even hold a candle to good old AppleWriter. Sigh.

Let's move on to a question of great import . . .

Do People Actually Enter Those Stupid Contests of Yours?

Sure thing. In droves, even. In fact, its gotten so bad that even some of the industrial strength tinajas might have to go on allocation.

Our latest tinaja quest winner is Bob Smith from up in Colorado. He has a paper jogger you can build up from a cheap sander. More details on this in a future issue. Bob also does advise us that the *Xerox* thermal transfer material for T-shirts is being phased out, but that the new *Sharp* color copiers do offer a similar process and material.

He, along with some other readers, feel that the *Sublicolor* T-shirt stuff, while workable, is based on a very ancient duplicator technology and is insanely overpriced, particularly for the ongoing materials and supplies.

Several contest entrants put me onto what appears to be the best buy in a clamping guillotine style paper cutter. This is the *Martin Yale* Model 7000E, which lists for $499, but is dealer priced at $299 in lots of two.

A sturdier and much more rugged machine is shown in the unusual *Kelsey Company* catalog, while some of highly overpriced also-rans appear in the *Wolf Camera* and the *Clark Associates* catalogs, as well as in the *Printing Impressions* trade journal.

It also seems that there is a glut on the market for used, old fashioned guillotine cutters. Apparently, what you have to do to get one is ask any specialty paper wholesaler, since all of the specialty mills have gone to newer and more expensive cutters.

My own solution was to approach a local school and offer to sharpen and rebuild a broken old cutter of theirs, in exchange for its free use. This cost was all of $20, and the final appearance of all my self-published books has now gone up sharply.

A cut above, you might say.

The entries in our "neat stuff" contest are still pouring in. Tellyawhat. To turn all of this into an ongoing adventure for you, I'll just give you the answers here. Write or call all these people and see what you get – I guarantee you will be absolutely amazed: *Bizarro Inc.*, *C and H Sales*, *Hank Lee*, *Jerryco*, *Kelsey*, *Lindsay Publications*, *Meredith Instruments*, *Small Parts*, *Synergetics* (sneaky, eh what?) and, of course, the good old *Whole Earth Review*.

How do I get Started Programming a IIgs?

Good question. There are at least three different levels that you can use when writing machine and assembly language code for the IIgs.

The simplest level is to use the tools and resources you already did when you were back IIc and IIe programming. The intermediate level is to improve and upgrade these tools to include such things as the new 65816 instructions and all the IIgs toolbox calls. And, the highest level is to go and use the ultimate weapons along with the full blown IIgs development environment with its fully relocatable code. APW is good for this.

The irony of all this is that it is not at all clear which level will give you the best final results. To play the IIgs game "their way" using the ultimate weapons involves a lot of learning, a ridiculous amount of time, plus an awful lot of frustration. Worse yet, many of the toolbox routines are inexcusably and intolerably slow, and can in no way compare to the hand crafting of fixed position traditional IIe-style code.

Examples of this sort of thing are the serial interface circuitry, parts of which are thirty times slower than on a IIe; AppleTalk firmware and protocols that slow down what should be a 240K baud rate to much less than plain old 9600 baud; and the text and screen tools that are just plain too slow or too marginal for serious word processing uses.

At any rate, let us look at some useful tools and resources. Figure one gives you a partial listing.

I have always felt you should hand code a few hundred lines of machine language before you even think about going to an assembler or any more powerful tools. This is the only way to fully and totally understand such things as addressing modes on their most fundamental level.

Some of the traditional and older programming tools include Apple's own EDASM assembler, which can be helped along bunches by using AppleWriter for "new way" power editing. I like to think that my own *Micro Cookbooks*, volumes I and II, and my own *Apple Assembly Cookbook* are also both very useful and solid on fundamentals.

As additional resources, the *S-C Software* people have an interesting and useful assembler. They also do stock many of the popular third party assembly books and utility routines.

Their *Apple Assembly Line* publication has secret insider IIgs info in it that's not to be found elsewhere.

Two IIgs specific books that are in the "must have" category are Gary Little's new *Exploring the Apple IIgs* and Eyes and Lichty's *Programming the 65816*. A.P.D.A. has them.

From Inside a machine or assembly language program . . .

```
CLC  XCE  –  To get into the native mode
SEC  XCE  –  To get into the emulation mode

REP  #$10 –  For an 8-bit accumulator and memory
SEP  #$10 –  For a 16-bit accumulator and memory

REP  #$20 –  For 8-bix X and Y registers
SEP  #$20 –  For 16-bit X and Y registers
```

From the monitor Lister . . .

```
 1=m  –  For 8-bit accumulator and memory
 0=m  –  For 16-bit accumulator and memory

 1=x  –  For 8-bit X and Y registers
 0=x  –  For 16-bit X and Y registers
```

Fig. 2 – How to change the IIgs operating modes.

The current Apple thinking is to recommend doing all of your high level programming in the "C" language and to use an assembler called *ORCA/M* for all of your tightly linked machine language modules.

One very big advantage of the "C" language is that it links beautifully to custom machine language code.

The foremost source for most IIgs programming books and software are the *A.P.D.A.* people. They offer a programming system called APW., and short for *Apple Programming Workshop*. This is a series of shells, assemblers, linkers, debuggers, and assorted goodies that give you a total IIgs programming environment.

We will be seeing lots more on APW. in future columns. Two other essential books for IIgs programming include the *IIgs Firmware Reference Manual* and the *Apple IIgs Toolbox Reference*.

What are the IIgs Operating Modes?

One of the real surprises when you try to list a IIgs file or program is that you might get some very strange garbage instead. Other times, you might get a listing that is correct for a while and then suddenly gets trashed.

What is going on here?

First, note that many on-disk IIgs files are special self-relocating *load files* of system types $B3 or $B5. The difference is that the $B3 files will usually stand alone, while the $B5 types are intended to run under a shell program of some sort. Full details on both of these file types appear in the *Apple IIgs Programmer's Workshop Manual*.

Second, and even more fundamental, there are several new operating modes that are involved in the IIgs. Figure two summarizes these.

There is a *native* mode in which the 65816 acts as a full blown 16-bit microprocessor, and an *emulation* mode in which the chip thinks it is a plain old 6502 or 65C02.

You switch between the emulation mode and the native mode by flipping a special flag that is "hidden" behind the carry flag. As figure two shows us, the *XCE* command will automatically exchange the carry and the emulation flags. On reset, the 65816 always will power itself up in the 6502 emulation mode.

In the emulation mode, all memory references, all accumulator loads, and all index registers are supposed to be 8-bits wide. In addition, many of the fancier 65816 instructions might be meaningless or else might do bizarre things. This 8-bit emulation mode is primarily intended for running older software as written originally for the 6502 or 65C02.

In the native mode, you have your choice of 8-bit or 16-bit memory references and separately of 8-bit or 16-bit sizes for those X and Y registers. As figure two shows us, these selections are pretty much independent. The SEP and REP commands are used to set the "m" and the "x" flags.

Note that for most uses most of the time, a 16-bit microprocessor usually makes mostly 8-bit commands and primarily carries out 8-bit operations. To not do so is often a waste of processor time and of memory space. It is only when you specifically do need "wider" words that you will switch into full 16-bit operation.

For instance, one ASCII character in a word processor often only has to be 7 or at most 8 bits wide. There is no point whatsoever in using a full 16-bit word here; to do so would most likely slow you down and certainly would double the length of all of your word processor text files.

Since you now have a choice of memory and accumulator size and a separate choice of index register widths, a semi-intelligent lister (such as the "L" command in the monitor) gets confused if it does not know what you had in mind when it starts doing its listing.

Thus, and again per figure two, you use the monitor 0=m or 1=m commands tell your lister whether it is to expect any 8-bit or 16-bit memory and accumulator references. You separately use the 0=x and 1=x commands to tell your lister whether it is to expect either 8-bit or 16-bit widths for your X and Y index registers.

Note that both the "m" and the "x" here *must* always be in lower case.

For instance, a code sequence of $A9 $06 $EA will be listed as LDA #$06 and a NOP if you are in the 8-bit mode (1=m) and will be listed as LDA #$EA06 if you are instead listing in the 16-bit mode (0=m).

Depending on where you are in your program, one and only one of these will be correct, and if you do select the wrong one, you will get meaningless garbage.

```
% drawapples - draws open and closed apple symbols.


% These symbols are registered trademarks of Apple Computer.

% Consult them directly for legal use restrictions.


/openapple {gsave /fontsize exch def /Symbol findfont [fontsize

0 0 fontsize 0 0] makefont setfont moveto (\360) false charpath

clip fontsize 12 div setlinewidth stroke grestore} def


/closedapple{gsave /fontsize exch def /Symbol findfont [fontsize

0 0 fontsize 0 0] makefont setfont moveto (\360) show grestore} def


% demos - re[illegible]re use . . .


/xposition 200 def

/yposition 300 def

/fontsize 60 def
```

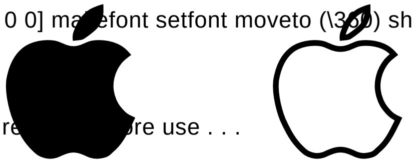


Fig. 3 – Postscript code for open and closed apples.

Finally, remember that any lister will lie like a rug unless it begins its listing process at a valid point in legal and legitimate code. In the IIe, you had a hint in the way of lots of question marks when something was wrong with your listing entry point.

On the IIgs, since each and every op-code is used for something and lists as such, it takes a little more practice and experience to separate valid listings from files, wrong starting points, or outright trash.

How can I Print Directly Onto Metal?

As usual, you start out with *Applewriter* and then create your original artwork on a *LaserWriter*, ending up with a negative transparency. Then you go to yet another little known and magic material that has been scunging around for dozens of years. This one is called *Metalphoto*.

The Metalphoto products are a series of aluminum sheets that have gone halfway through an annodizing process and have then gotten suitably photosensitized. You expose and then develop the sheets to create a real photographic image. Finally, you finish off the annodizing process by boiling the plates in special glop. The glop seals up the formerly open or "sponge" surface of the aluminum and creates a smooth, sapphire hard, and quite transparent aluminum oxide overcoat that will protect and "lock in" the photo image. The image gets actually locked "inside" of the aluminum, rather than on its surface.

While you can in fact damage a Metalphoto dialplate by physically gouging it out, it is almost totally resistant to heavy wear, handling or weathering, most industrial solvents, vandals, and many chemicals.

Metalphoto is available in several sizes, thicknesses and colors, ranging from foils to plate. One single 3 x 5 inch dialplate may cost you around $5 using this process.

With a normal exposure, you'll get black over color. If, instead, you use a negative image, you will get colored lettering over a black background.

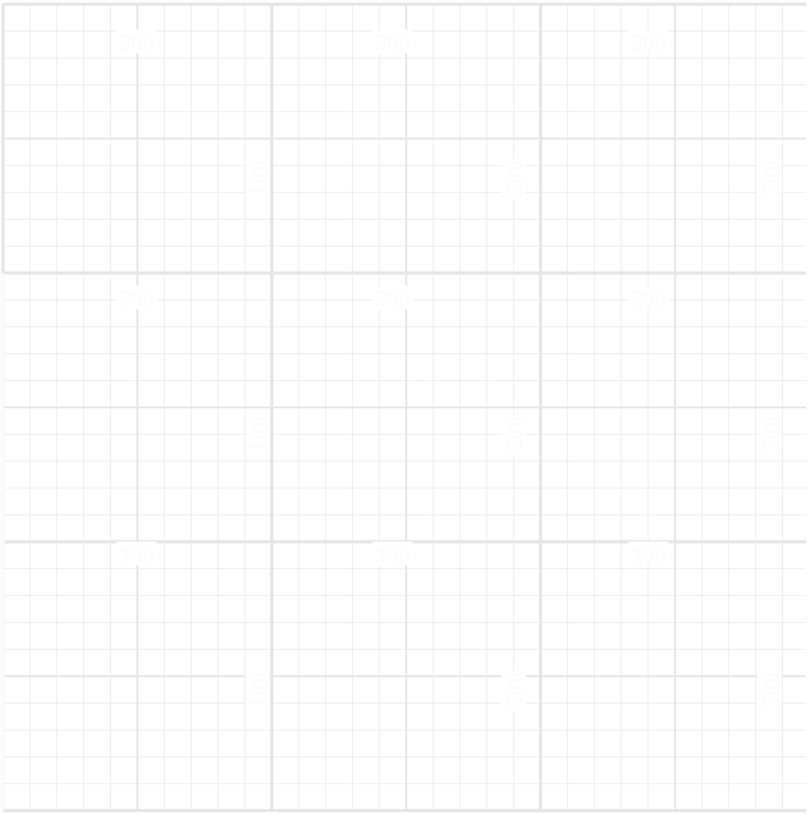


Fig. 4 – A typical PostScript standard point layout grid.

The uses are obvious: signs, logos, nameplates, dials, etc... The Metalphoto salesmen even use aluminum business cards with their own personal halftone photo on them. The museums and exhibit, park and monument people all like Metalphoto because it is reasonably vandal and weather resistant.

Back in the golden age of hacker electronics (check out *Popular Electronics* in 1967 to 1970), I did use Metalphoto for dozens of different project dialplates and even had a custom panel service going.

A competing and apparently less popular process is called *Fotofoil*.

Tellyawhat. For this month's contest, just dream up an off-the-wall, new, or obscure use for Metalphoto or Fotofoil. We will have the usual books and tinaja quests as prizes.

Tell me More About IWEM.

The IIgs disk file /SYSTEM.MASTER/APPLETALK/IWEM on the IIgs system master disk is an ordinary textfile that contains an excellent imagewriter emulator for your LaserWriter or for pretty near any other PostScript speaking printer.

You can easily move this file over to a IIe or over to most any personal computer just by grabbing it. When transmitted as PostScript code, IWEM first tests for its own presence inside your printer and then downloads a persistent clone of itself that remains so long as power is applied.

To switch into your imagewriter emulation mode on a IIe or an IBM clone, you enter the code "_WBJ_". The rest of that particular job will be done as if you had a high resolution version of an ImageWriter printer.

The code is very easily modified to change it back into an emulator of most any daisywheel or dot matrix printer you care to. You might also select proportional type fonts, but note that this works best with a left justified text and can cause all sorts of serious hassles with spreadsheets or columnar data.

As we found out last month, the present LaserWriter Appletalk driver automatically activates your IWEM code whenever port seven gets initialized and then trys to force you to send all your code in the emulation mode. Thus, you will get PostScript listings rather than postScript code printed out, unless you make some changes real quick like.

To beat this, you simply replace IWEM with any code variation you like while obeying two rules: (1) The final code must be named IWEM; and (2) there MUST be a procedure somewhere near the start of IWEM that is defined with the name _WBJ_.

To review what happens in the IIgs, with an AppleTalk installation, IWEM is persistently downloaded to your PostScript speaking laser printer, if it is not already there. Later on, PR#7 or another AppleTalk initial activation, a single "_WBJ_ command is sent out. This turns on the imagewriter emulation mode and locks you out of PostScript. Unless, of course, you properly rearrange the scenery to suit yourself.

IWEM apparently *demands* low ASCII input, with bit 7 always a zero. Thus, ProDOS AppleWriter 2.0 will work, but 2.1 may not. Similarly, a simple "force low ASCII" machine language driver should be used between Applesloth and IWEM. This can be done from inside BASIC with a few POKE commands, and is left as an exercise for all of you more serious students. So there.

What is this Month's PostScript Utility?

How about a pair of them? Figure three shows you all of the sneaky tricks needed to print the open and closed apple symbols as you might need for an Apple book or software manual or whatever.

While the *Symbol* \360 character prints as a solid apple on all LaserWriter versions, it may or may not on competing brands of PostScript printers. So, this just might end up as machine-specific code.

Figure four is a standard point grid that is most handy for layout work. One possible listing of PostScript code for this appears in figure five.

The grid can be shown either in a black or in a light gray. I've shown you the gray version here. The obvious advantage of gray is that you can overwrite it with all of your final real PostScript images.

The code of figure five can fill an entire sheet. I've only shown you the lower left corner of the grid in figure four. By the way, it usually will pay to offset your grid into the printable area of the sheet, putting the grid origin at 30, 30 and then doing a *30 30 translate* before you start your PostScript work. This will both look better and give you fewer suprises.

You can also xerox onto sheets with preprinted layout grids on them. That can greatly simplify such things as curve tracing, digitized signatures, custom logos, and stuff like this.

While the rubbergrid we looked at previously can give you "perfect" and single pixel dot crossings, this true standard point gray grid apparently can not, since an exact size has to be maintained. The black grid does not share this problem. The gray letters are also inherently less sharp than are the black ones.

Or at least I don't know how to correct either of these two "features". Sorry about that. If you have a fix, please let me know. Possibly a very special spot function can help here.

A centered point grid is simiarly possible, and is handy where layout symmetry is important. We'll pick up code on this some other time.

As per usual, this is your column and you can get technical help per the end box. Yes, we do have reprints of all previous columns available. Call or write for further info and for a copy of the new free stuff list.

```
% regpointgrid - Full sheet gray point grid with lower left origin.
% . . . . . . . .

% Copyright c 1988 by Don Lancaster & Synergetics, Box 809,
% Thatcher AZ, 85552, (602) 428-4073. All rights reserved.
% Personal, non-commercial use permitted so long as this
% header remains both present and intact.

/setgrid { save /rubbersnap exch def /size exch def translate size
dup scale} def

/drawlines {72 300 div lw mul size div setlinewidth /hposs 0 def
#hlines gs div 1 add cvi { hposs 0 moveto 0 #vlines rlineto stroke
/hposs hposs gs add def} repeat /vposs 0 def #vlines gs div 1 add
cvi {0 vposs moveto #hlines 0 rlineto stroke /vposs vposs gs add
def} repeat} def

/showgrid{ seegrid {gsave /#vlines exch def /#hlines exch def 106
45 {pop pop 0} setscreen 0.9 setgray

/gs 1 def /lw 1 def drawlines fat5 {/gs 5 def /lw 3 def drawlines} if
fatter10 {/gs 10 def /lw 5 def drawlines} if grestore}if} def

/fat5 true def /fatter10 true def /seegrid true def 0 0 10 setgrid 60
78 showgrid clear rubbersnap restore

135 25 {dup mul exch dup mul add 1.0 exch sub} setscreen
0.99 setgray

/erasebox {gsave currentpoint /yc exch 1.5 sub def /xc exch 1 sub
def xc yc moveto 0 9 rlineto 15 0 rlineto 0 -9 rlineto closepath
1 setgray fill grestore}def /Helvetica-Bold findfont 8 scalefont
setfont /numberow {/numnum exch def /numpos exch def /xrun
43 def 9{xrun numpos moveto erasebox numnum show /xrun xrun
100 add def} repeat} def

97 (100) numberow 197 (200) numberow 297 (300) numberow 397
(400) numberow 497 (500) numberow 597 (600) numberow 697
(700) numberow 90 rotate -103 (100) numberow -203 (200)
numberow -303 (300) numberow -403 (400) numberow -503 (500)
numberow -90 rotate 406 725 moveto 20 setlinewidth 180 0
rlineto stroke 1 setgray /Helvetica-Bold findfont 13 scalefont
setfont  410 721 moveto (Normal Grid { 0 0 translate }) show

showpage
```

Fig. 5 – Postscript code for the standard point grid.

Don Lancaster’s

ASK THE GURU

March, 1988

Perspective drawing
Help on the Apple III
Self-publishing secrets
Alternate IIgs monitors
INH problems on the IIgs

Apparently the powers that be at *Apple Computer* have belatedly discovered that by far the most absurdly ludicrous words ever to be uttered by mankind are "See your local Apple dealer for technical assistance." At long last, Apple is now making a really major commitment to improving and upgrading their end-user technical services. Expect a direct tech help hotline soon, along with much wider and easier *AppleLink* BBS access, plus several new user group services.

Now, if we could only get them to loosen up on their repair and service information, we might actually end up with something workable.

Apple also provides a new and free *Connections* guide, a handy resource directory for disabled children and adults. This is available through their special education office.

At the same time, Apple has now "improved" their AppleTalk connectors so they do not fall apart quite so easily. Naturally, they substantially upped the price when they did this.

What they really need to do with AppleTalk, of course, is make it so it inherently works with either doorbell wire or telephone twisted pair. They should also get rid of all of that ridiculous AppleTalk software overhead that makes plain old honest 9600 baud serial communications significantly faster for many users much of the time.

Rumors are surfacing of an Apple developed genlock video board that plugs into a IIe or a IIgs, and provides an overlay of IIgs graphics onto an external true NTSC video signal.

Apparently, the board is powerful enough that it can provide full IIgs graphics even when plugged into a IIe that has not been upgraded!

More on this as the drama unfolds.

I have felt for years that the first and foremost decision that should be made when designing a new personal computer is when and how all of the NTSC frame grabbing and genlocking will be done. After that gets decided, then, and only then, should you pick a CPU, ROM, RAM, and all the other frivolous addons that surround your fundamental video interface.

There have long been rumors of a "super" version of AppleWriter that handles a 256K work file, includes a spelling checker, and provides other assorted goodies. Well, I have finally tracked this dude down. It was first known as *Super Applewriter III*. Just as the Apple III was being flushed, a few bootleg copies were released to user groups by Apple.

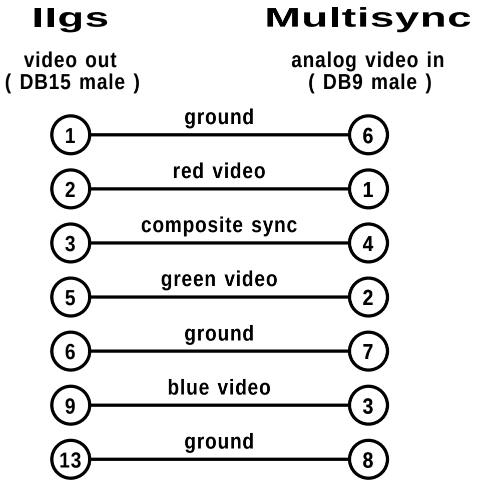


Fig. 1 – IIgs video cable for the NEC Multisync monitor.

Chances are you can get a copy if you do ask the right III hacker. To snoop, start from ProDOS, insert the SOS disk and then you do a BLOAD SOS.INTERP, A$57F2, T$0C.

At any rate, it turns out this code was older and less powerful than AppleWriter 2.0 or 2.1, except for that larger workfile and the new spelling checker. The memory was handled in banks of 32K each, and you could select as many as you needed. The bank switching was apparently handled by the III operating system.

Today, though, it is probably far better to choose a standard spelling checker. It would also be faster and better to rewrite the 2.0 or 2.1 memory access code to use all the more powerful IIgs commands.

I guess it would be heresy to leave the advertorial out of this space. So, do not forget you can now get bound reprints of all of my *Ask The Guru* columns. And, I do have lots of great PostScript stuff for you when you call or write.

The insanely stupendous miracle fantasmagorical breakthrough for this month involves doing true perspective drawing out of AppleWriter that includes accurate circles and arcs, and – believe it or not – genuine perspective lettering.

More on this shortly. But first ...

Which Monitors Work With the Apple IIgs?

We saw way back in the May 87 column just how to use that *Sony* KV1311CR receiver/monitor with a IIgs. This is still my favorite and the one I use. Besides ending up $100 cheaper than the Apple color linear RGB monitor, this one also is a 196 channel tv set with a remote control, and has lots of other neat features.

Two other linear RGB monitors that are useful with the IIgs are the *NEC Multisync*, and that *Magnivox RGB-80*. Figures one and two will show you the cable connections to use either one of these alternate monitors. And, if you do have a favorite monitor of your own, be sure to send your circuit in so we can share it with the other *Computer Shopper* readers.

What is the INH Problem On the Apple IIgs?

Basically, there is a really major screwup in the Apple IIgs hardware that makes use of the INH inhibit line by a plug-in card difficult and sometimes impossible.

What happens is that INH will only work in certain memory ranges, and will cause a memory contention fight in others. Specifically, some real bad memory contention (and a system blowup and possible damage) might happen if you do lower INH while addressing $6000-9FFF in main memory, or if you address $0000-5FFF or $A000-FFF in auxiliary memory.

It is possible to destroy the MEGA II chip if you try this. More details on this in the IIgs Technical Note #32, from your local user group or BBS.

Where can I get Apple III Help?

Outside of it being obscenely overpriced, monumentally mismarketed, poorly assembled at first, and woefully underdocumented, the Apple III wasn't all that bad a machine. The III was to be that great IIe killer, along with Lisa and a few other machines whose names I don't recall.

In fact, the III still does see lots of use in community college administrative work, and every once in a while, one will show up at a bargain price. Their *3 Easy Pieces*, a forerunner of *AppleWorks* does remain a useful integrated word processor, spreadsheet, and data base.

I get a suprisingly high number of helpline calls on the Apple III. It turns out there is a gang called *On Three* that publishes a newsletter, and offers a number of support and networking products. If you are using or going to use one of these machines, be sure to check them out.

Tell Me All About Self-Publishing Books

As we found out in our previous ATG columns, custom laser book printing is more than cost competitive with jiffy printing, escpecially when you do not know exactly how many copies you will actually sell.

Other major advantages of laser self-publishing are that you can make additions or corrections at any time, you can send out your review copies within minutes after the manuscript gets in its final form, that you can instantly and electronically transmit copies anywhere in the world, and that you can print each customer's name in gold on each copy.

But the biggest advantage of all to laser self-publishing is that everything you do is wid your own widdle hatchet. There's no way that others are able to screw up the works for you through any incompetence or delays, sloppy work, lack of attention to detail, misinterpretion, outright lies, committeespeak, missed delivery schedules, or overpricing.

You instead get to do all of these things all by yourself.

For nearly a year now, I have been self-publishing my *Ask The Guru* reprints, and I thought I would bring you up to date on how this is being done. The bottom line is that the process works and works very well, and that I intend to be publishing many other titles by this method.

At present, the 180 page (90 sheet) books are produced chapter by chapter in lots of twenty. This limit is set by the LaserWriter's paper tray capacity. The AppleWriter textfiles all reside on four 3-1/2 inch IIgs disks.

The WPL supervisory language is used to automatically print a single chapter at a time, with a high school freshman keeping an eye on the more or less automatic process while she is collating or binding.

Printing is done on a high bulk 5 mil *Husky* offset paper that is very attractive and resists print-through better than copier paper. Eventually, when the price comes down, I will switch to an ultra-white custom laser paper. Toner cartridges are refilled up to seven times at a cost of less than $7.50 per refill, which drops the toner cost under 0.4 cents a page.

The covers are printed on heavy *Skytone* parchment and then are run through a *Kroy Kolor* fusion machine to improve the toner's durability by Bakerizing. A wrap-around *Unibind* clear plastic thermal binding is used as an outer cover, completely protecting the cover toner from scuffing or wear. A special homemade jig is used to guarantee the covers are square.

An alternate cover process that I use on my smaller booklets is to laser print and then *Kroy Kolor* onto an attractive cover stock. This is followed up by a protective lamination of clear *Kroy Laminating Film*.

After the jogging, collating and a thermal binding, the books are given a trim with a guillotine paper cutter. This gives a professional final edge.

Naturally, I'd be most happy to send you a sample copy of all this at the going rate. Do write or call if you are at all interested.

So, What is Next?

The next really big step in the self-publishing process is *publishing on demand*. At the press of a button, out should pop one complete and personally customized book copy.

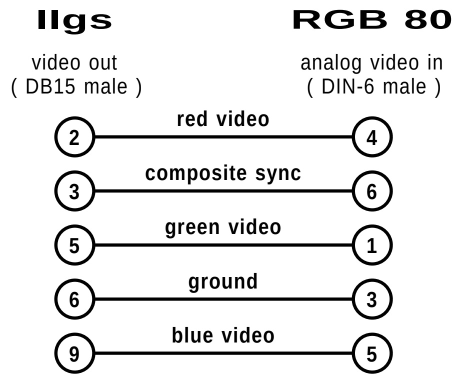


Fig. 2 – IIgs video cable for the Magnivox RGB 80.

All collated and ready for binding. Fully unattended, except possibly for a page flipover halfway through.

To do this will require a dramatic speedup in the PostScript processing and communicating time. By use of all my compiling utilities, I can now print a three column, ten font, two illustration plus header sheet using a mere 14 seconds of pre-print processing time.

By going out the IIgs game paddle port at 57600 baud, I should be able to communicate well over 7 times faster than AppleTalk can. A hard disk can further speed up the WPL access of all the individual page files. And I should have a new LaserWriter accellerator on hand "real soon now".

An intermediate objective of 12 seconds per printed page should be reachable. This would translate to around 2160 seconds per book, which is roughly one book produced every 35 minutes. Note particularly that any and all of those minutes could be done during lunch breaks or at night.

If you project what a laser printer will be capable of doing a few years from now, this is obviously the time to be getting into and debugging your self-publishing act in a big way.

Opporknockity tunes but once.

How can I Make a Perspective Drawing?

Why, with AppleWriter on a IIe or IIgs, of course. How else could you possibly do a first quality perspective drawing?

The PostScript language and a laser printer does help out a tad along the way. Check out figures three and four. What you have here is a fast, simple, and disgustingly powerful means of creating your own custom two-point perspective drawings.

Note particularly that each and every object in the perspective grid can be individually rotated; that very accurate perspective circles and arcs are trivially included; and, above all, that *full perspective lettering* is very simply and easily done, using most any font of your choosing.

And, this is by no means a "static" display system. You can easily do a zoom, a reposition, or even a sequential animated flyby with only a very few extra keystrokes. Not too shabby for AppleWriter, eh what?

Figure five shows you some of the PostScript code involved. The actual two-point perspective math is not all that hairy, and we may get into it in a future column. All you really have to do is "ray trace" a pair of similar triangles to locate the end points on a *picture plane* that lies between the subject and the observer.

Three global parameters are under your control. The XO value is how far to the right or left of center the observer is sitting. The YO value is how far in front of the picture plane the observer is sitting. Finally, the ZO value will be the height of the observer's eyes above or below the "ground" plane.

Now, it is important to remember that two-point perspective, like any attempt whatsoever in showing a 3-D world on a 2-D sheet of paper, is an illusion and an approximation.

Artists usually recommend a 5:1 to 15:1 ratio for YO, or the illusion may fail. Put another way, you usually want to be five or more times the distance away as the object is wide. Architects will prefer fifteen times. Thus, fairly large values of YO will often give you the most realistic final results.

Similarly, an architect will almost always use an "eye level" for his ZO, and very high or very low views will be unrealistic. Thus, you will most often use a fairly low value for ZO. Remember that all the vertical lines will remain vertical in any two-point perspective drawing.

Your fourth control parameter is called OBJROT. This one lets you rotate any individual object on the ground plane. Think of the ground plane as a giant sidewalk. Any of the individual objects on the sidewalk can be rotated towards or away from you, creating all the more interesting angle views.

You can move the observer around simply by changing the XO, YO, and ZO parameters. The subjects can be made more or less interesting by a change of OBJROT. A custom proc that is similar to the stock PostScript rotation matrix is used to accomplish these rotations.

As figure five shows us, there are both absolute and relative moves and draws. All of the relative moves will *include* any requested rotation; those absolute ones will not. You should thus use only absolute moves and draws to position an object, and then use relative moves and draws for all the rotatable details of that object.

Curves and arcs are handled by using my curvetrace utilities. To do a circle, the correct locations of 13 *intermediate points* are calculated, one for each 30 degrees of arc. Each of these intermediate points is then converted into a *pair* of coordinates that are a very small angle *before* and

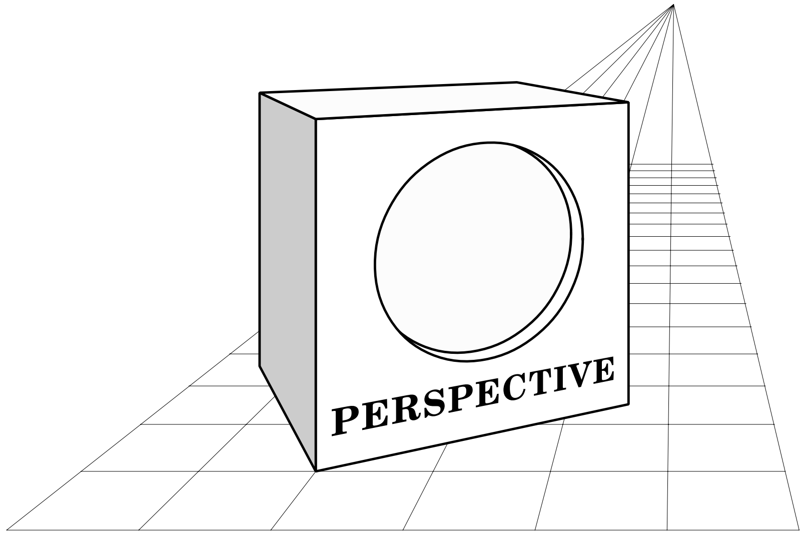


Fig. 3 – AppleWriter strikes again . . .

beyond the desired point on your selected curve.

The four resulting data values are then converted into an X position, a Y position, and a direction vector. These are then used as part of a continuous curvetracing process to generate your desired circle or arc.

The results are much better and much more accurate than the phony elliptical approximations that were previously used by draftsmen, except for the most severely squashed of circles. The really squashed stuff can be easily handled by adding a few more intermediate points.

The lettering is done with my pixel line remapping routines. The message gets scanned exactly one line of vertical pixels at a time. Each pixel line is then perspective transformed and height scaled so it drops into the proper place at the proper size in the final image.

Yes, as the image rotates, zooms, or expands, the lettering will automatically adjust itself to give the correct result. There's no limit to your choice of fonts. In fact, anything that you can show "flat" on any PostScript page can be "pasted" onto any side of any perspective object in your image. Anywhere.

The PostScript processing speed is amazingly fast for most perspective drawing, although things will slow down on an irregular clipping boundary. These already fast times can be much further sped up by compiling or pseudo-compiling, done with my PostScript utilities.

That pixel line remapping involved in the perspective lettering might become excruciatingly slow for large letters or long messages. But, as per usual, there's that good old "Uh, compared to what?" factor.

The perspective techniques shown here, can, of course, be ported over to most any personal computer. All you will really need is your favorite word processor or comm program.

These routines can all be extended just for you on a custom basis. Write or call if you are interested in this or any of my other PostScript goodies. I have got neat stuff not available elsewhere for everyone from a bare novice to the most gonzo of you advanced Postscript developers. Let's hear from you.

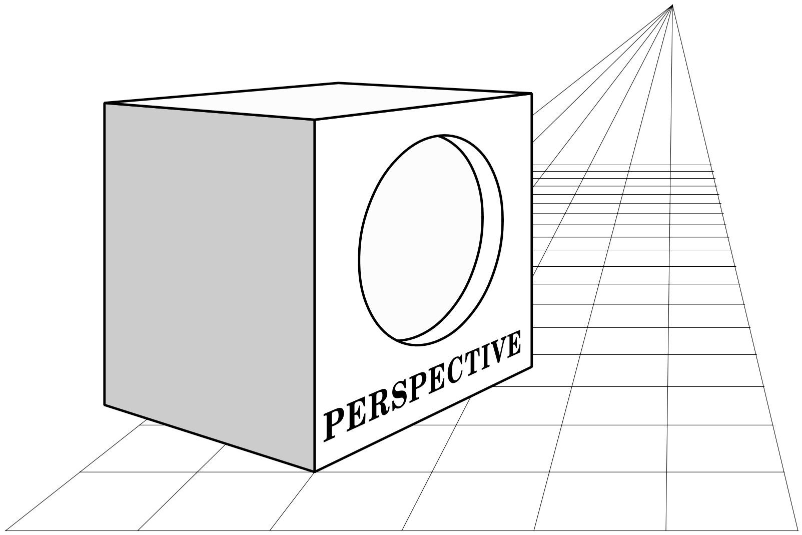


Fig. 4 –   . . . And, two keystrokes later, once again.

```
%  Copyright c 1988 by Don Lancaster & Synergetics, Box 809, Thatcher AZ, 85552.
%  (602) 428-4073. All rights reserved. Personal, non-commercial use permitted so long as
%  this header remains both present and intact.

% perspective transform - converts x,y,z into X and Y on stack:
/px {/zz exch def /yy exch def /xx exch def yo dup yy add div xx xo sub mul yo dup yy add div
zz zo sub mul} def

% save and restore previous x,y,z position:
/psave {/zh zz def /yh yy def /xh xx def} def /prestore {/zz zh def /yy yh def /xx xh def} def

/pm {px psave moveto} def % perspective absolute move  x,y,z and hold

/pd {px psave lineto} def % pd perspective absolute draw x,y,z and hold:

% prm perspective relative x,y,z move with objrot rotation:
/prm {/zi exch def /yi exch def /xi exch def objrot cos xi mul objrot sin yi mul sub xh
add objrot sin xi mul objrot cos yi mul add yh add zi zh add px moveto psave} def

% prd perspective relative x,y,z draw with objrot rotation:
/prd {/zi exch def /yi exch def /xi exch def objrot cos xi mul objrot sin yi mul sub xh add
objrot sin xi mul objrot cos yi mul add yh add zi zh add px lineto psave} def

% default distances from observer to picture plane:
/xo 20 def            % left and right
/yo 80 def            % into picture; avoid small values
/zo 40 def            % up and down; avoid large values
/objrot 30 def         % relative xy object rotation

% --- demo.remove before use ---
300 400 translate   5 dup scale % this is the center horizon and the scale

% perspective grid
0 setlinewidth /startat 0 def 0 setlinewidth 2 setlinecap 19 {-30 startat 0 px moveto 30
startat 0 px lineto stroke /startat startat 10 add def} repeat /startat -30 def
7 {startat 0 0 px moveto 0 100000 0 px lineto stroke /startat startat 10 add def} repeat

106 35 {dup mul exch dup mul add 1.0 exch sub} setscreen  % a non-putrid gray

% a shaded perspective cube
2 setlinecap 2 setlinejoin -10 10 0 pm 0 0 30 prd 30 0 0 prd 0 0 -30 prd closepath gsave
1 setgray fill grestore 0.2 setlinewidth stroke

newpath -10 10 0 pm 0 0 30 prd 0 30 0 prd 0 0 -30 prd closepath gsave 0.6 setgray fill
grestore 0.2 setlinewidth stroke

newpath -10 10 0 pm currentpoint newpath moveto 0 0 30 prm 30 0 0 prd 0 30 0 prd -30 0
0 prd closepath gsave 0.99 setgray fill grestore 0.2 setlinewidth stroke

showpage
```

Fig. 5 – Some of my Postscript perspective routines.

Don Lancaster's

ASK THE GURU

April, 1988

A new high-tech toy
Perspective transforms
New LaserWriter details
IIgs AppleWriter patches
Toner cartridge reloading

Well, I still don't have my new *LaserWriter* yet. Word has it that there are some minor teething problems with the new production facility that is intended to churn these new machines out.

Just as soon as I do, I'll give you the full lowdown on printing speeds and whatever. Naturally, there will be fatal errors and other serious bugs on all of these new printers, so my *Computer Shopper* columns might serve as a clearing house and forum to let you know what these bugs are and how to get around them.

There are really three brand new LaserWriters, all of which share the same *Canon* SX engine. This engine is just about the best available today for low end laser printing. On the other hand, the per-page toner costs are still substantially higher than on the CX (more on this below), and the latest third-party CX toner refills are almost as solidly black as those you can get from the SX.

You can upgrade between any of the new machines by exchanging cards. There is now a several hundred dollar "penalty" to do so.

The low end machine is known as the *LaserWriter II SC*. In my opinion, this gutless wonder is a totally useless joke, since it does not speak the industry standard PostScript page description language. The -SC is beneath further comment.

The mid-range machine is called the *LaserWriter II NT*. This one is essentially an improved upgrade of the LaserWriter Plus, with pretty much all of the same features. You now have a more durable, smaller, lighter, and generally upgraded engine that holds more paper and feeds it more efficiently, handles envelopes better, and offers more solid blacks. You also get slightly more memory and modestly faster speed.

The high-range machine is named the *LaserWriter II NTX*. This adds more memory, a faster 68020 main processor, and a LaserJet emulator to the NT. The main benefits are faster operation and more memory to hold more downloaded fonts at once.

Instead of providing an internal hard disk, there's a SCSI port that lets you mix and match four external hard disk or CD ROM devices.

The pricing is enigmatic at the present time. The dealer margins are insanely high, which either means that (A) Apple has now been struck with a corporate wide fit of *generositius excessus*, or else (B) Their suggested list prices are meaningless.

If a seer wanted to dust off their crystal ball, though, they just might make the following predictions for the "low street" prices, once all the dust has settled: $3795 for the high end -NTX, $2650 for the -NT, and no market whatsoever for the -SC.

At this writing, the best and by far the easiest-to-get service and repair manual for the SX engine is available from *Hewlett-Packard* for $49 as their fine *LaserJet II Service Manual* part number 33440-90904. They offer overnight delivery, along with VISA and an 800 number.

This manual is essential for any intelligent use of any SX-based laser printer. It will be interesting to see how long Apple will continue to pay H-P a $49 rebate for each new LaserWriter they manage to sell.

Apple has also introduced a new MIDI musicial interface for both the IIgs and the Mac. The list price, including the needed cables, is $99.

The AppleTalk network has been renamed *LocalTalk* and the price of the cables and connectors has been raised. After all, the name change did have to be funded somehow.

Word has it that the *International Apple Core* is in very deep trouble. At least that's what several help line callers seem to think who have received neither product nor refund for their numerous and repeated efforts.

What started out as an outstanding idea – a club supergroup with a fine magazine, superb public domain disk releases, and great insider tech info – long ago got mismanaged by total incompetents that ended up woefully overpricing themselves right out of their market.

Here's a quick reminder about my

This patch is for ProDOS Applewriter 2.0 version AWD.SYS.
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
    version 2.0. Plainly label this disk FOR IIGS ONLY!

2.  Get into /BASICS.SYS.  Then CALL -151 to reach 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

7.  UNLOCK AWD.SYS

8.  BSAVE AWD.SYS, A$2000, E$6020, TSYS.

9.  LOCK AWD.SYS
```

Fig. 1 – IIgs patch for ProDOS Applewriter 2.0.

Hardware Hacker sister column to this one that you will find running in *Radio-Electronics* magazine.

And, yes folks, its advetorial time. Complete *Ask the Guru* reprints of most back issues are available, as are my brand new *PostScript Language Perspective Drawing Utilites*, and my good old *Incredible Secret Money Machine* book. And now, just as soon as we can get that MS-DOS user untangled from our stage curtain . . .

```
This patch is for ProDOS Applewriter 2.1 version AWD.SYS.

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

    version 2.1. Plainly label this disk FOR IIGS ONLY!


2.  Get into /BASICS.SYS.  Then CALL -151 to reach 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
7.  UNLOCK AWD.SYS

8.  BSAVE AWD.SYS, A$2000, E$6020, T$0C

9.  LOCK AWD.SYS
```

Fig. 2 – IIgs patch for ProDOS Applewriter 2.1.

Is this 1984 or 1988?

Funny you should ask that. Way back in 1984, when ProDOS was first released, it was completely obvious the IIe was a dead machine, about to be blasted out of the water by the Apple III. And, of course, if that did not do it, Lisa would stomp any of the remaining IIe dregs into absolute nothingness.

So, in a fit of stupendous generosity, ProDOS was designed to last for an ridiculously long four years. Enough room was set aside to hold the years 1984 through 1987 inside the code. So, on midnight, December 31st, 1987, everybody's ProDOS system backed itself up by four years and started repeating.

Which isn't quite as bad as some *Tandy* machines that literally self-destructed on that date, but still . . .

The cure is either very simple or insanely difficult. Just make backup copies of all affected software, and then replace the backup ProDOS file with a copy of the P-8 file on the IIgs system master disk that has been renamed ProDOS.

Be sure to have lots of fun doing this on any locked and protected disks from now defunct companies.

There are some subtle differences between P-8 and the older versions of ProDOS, but the chances are these will not be a big problem for you.

If you are using a clock card that links to ProDOS, you might want to contact the manufacturer or your user group for more details.

What is the the latest on the Tech Tidbits?

Well, they still refuse to admit that LocalTalk is not at all needed for the LaserWriter, and that it often slows you down bunches and will severely limit your choice of host computer. Nor do they admit that the insanely expensive LocalTalk cables can often be replaced by a $4 null modem cable or even by three pieces of doorbell wire twisted together.

And they continue to refuse to admit that AppleWriter runs just fine on the IIgs, once those three byte patches are made that I've once again rerun here as figures one and two.

Or, better yet, go back to using a Super Serial Card. That IIgs serial firmware creates far more problems than it solves. Many other programs will also work just fine with a Super Serial Card but will hang with the IIgs internal serial firmware. Just be sure to select "your card" from your control panel. This also will bypass many of the IIgs print buffering hassles and incompatibilities.

Outside of these many grevious problems, *Tech Tidbits* does seem to be very slowly cleaning up their act. Someday, eventually, they just might even become a useful resourse. Here are some excerpts from them . . .

The quickest way to identify the new IIc or new IIe ROM upgrade is to type a CALL -151 followed by a (!) exclamation point. If the (*) prompt changes to (!), then you have dropped into the mini-assembler and have your new chips installed.

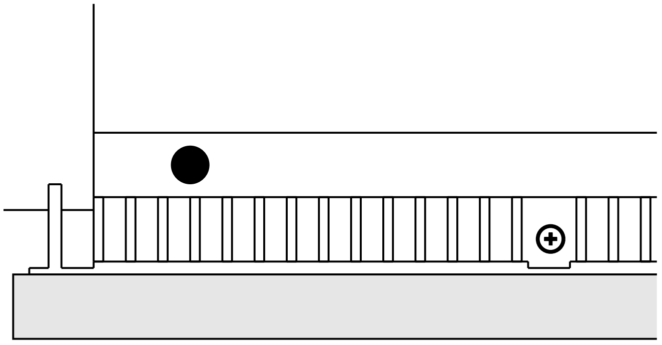


Fig. 3 – Adding an emptying hole to a CX toner cartridge.

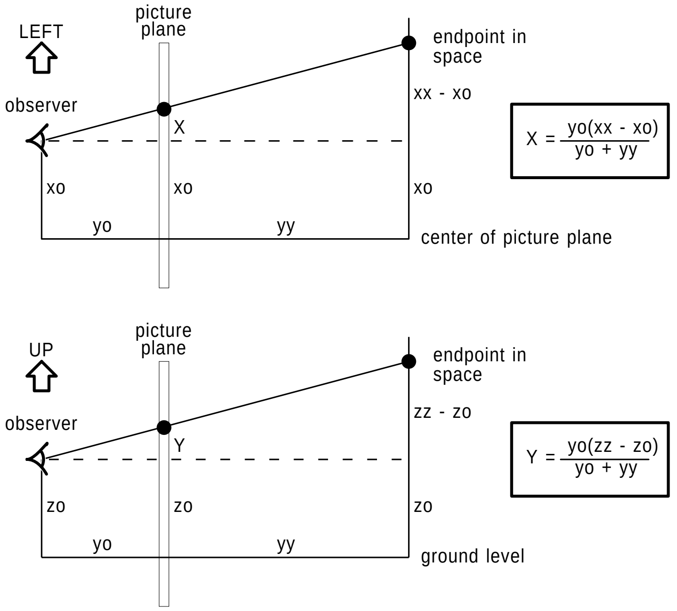


Fig. 5 – The 2-point perspective X and Y transforms.

One additional source for those LocalTalk connectors are the *AESP* people. *MDIdeas* has some true stereo sound add-ons for the IIgs.

SCRG sells adaptors that let you use an Apple Joystick with a II+, along with similar goodies.

Color laser toner is available from either *Michelin* or *Toner Technology*. But I have been most disappointed with any and all color toners that I have tried to date. They look awful and the pricing is outrageous. *Kroy Kolor* is often a better choice.

The recommended RAM expansion chips for the IIgs are the 256K bit dynamic parts organized as 256K x 1, with CAS before RAS refresh and a speed of 150 nanoseconds.

If you must use LocalTalk with the LaserWriter on the IIgs (it is ridiculously faster not to), be sure to use the launcher after the chooser, and be sure you can print to slot seven. Any reset or power down will disconnect the LocalTalk firmware.

Two of the many products that convert PFS files to Appleworks are available from *Jim Luther* and from *Sun Microsystems.*

Any New Hi-tech Toys?

It is a cross between an ant farm and an hourglass, and consists of nothing but some air, some colored water, and two shaded sands of differing densities. But it will slowly and stunningly create an incredibly realistic Arizona high desert landscape. Complete with all the mesas, buttes, cattle tanks, dry washes, and canyons. Even the block faulting and the *bajadas* of the basin and range geological province.

The effect is greatly enhanced when viewed along with some new age music and a suitable selection of choice herbal cupcakes.

Out here, they are available just about everywhere. A typical model is the *Whispering Sands* model #AXP-234992-B, available through Christopher Paul at *Alpha Centari.*

The physics that's involved gets suprisingly complex. When initially turned over, bunches of the two sands will drop and create an undulating foreground.

After a while, a nearly complete but crucially imperfect row of air bubbles will form, supporting almost all of the remaining sand. All the individual bubbles then "modulate" the falling sand into extremely fine and slowly moving streams, creating all the random mesas and buttes.

The only thing wrong with the beast is that there is no simple way to save an image once it is created. So, what I would like to do is find some way to use AppleWriter to do the same thing. Possibly helped along by making full use of PostScript.

So, for this month's first contest, just show me any old way at all to computer model what is going on here. If you can work AppleWriter in, then so much the better. There'll be an *Incredible Secret Money Machine* for the top twenty entries, and an all expense paid (FOB Thatcher, AZ) *tinaja quest* to the very best.

Give me an Update On Toner Cartridge Refilling

As we've seen many times before, those toner cartridges used on the *Canon* CX and SX laser printers are easily refillable. And the latest of third-party toner refills are now both super black and super cheap. Which completely turns around both laser printing economics and appearance.

For instance, a stock CX cartridge refill currently costs as much as $125

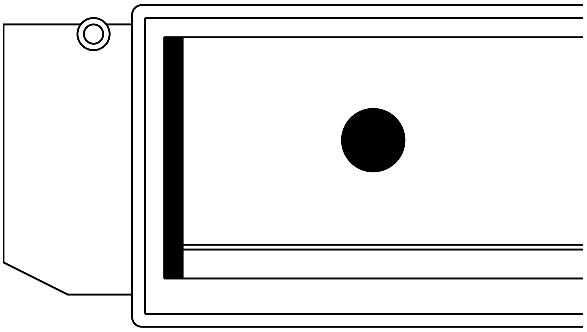


Fig. 4 – Adding a filling hole to a CX toner cartridge.

and is only good for 2500 copies. That is a nickel per page, just for toner. On the other hand, if you buy an empty but not previously refilled cartridge out of your sunday paper, it will cost you $5 in large towns and $10 in a smaller one.

With care, you can refill this used cartridge five times with toner refills priced as little as $7.50 each. Which costs out to only 1/3rd of a cent per page. That's less than jiffy printing, and only 1/15th the cost of always doing things "their way".

In fact, there is no point in *ever* buying a new CX cartridge, since the cartridge does not even start to get properly conditioned until 3000 copies or so, and since you might as well have somebody else use up all that grayish original toner for you.

The best source for refill supplies that I have found is Arlin Shepard over at *Lazer Products* in Colorado. And an independent toner testing and evaluation service is now available by way of *Thompson and Thompson*. There are lots of other suppliers, so call me if you want to know more about any of them.

There are two stages to the toner refilling. You first have to modify the cartridge once. After modifying, you can easily refill as many times as the drum condition lets you.

Figures three and four show you how to prepare your CX cartridge for reloading. You first melt a 1/4 inch hole in the spent toner holding tank as shown, by using a small soldering iron. Then you melt a 1/2 inch hole in the fresh toner supply tank. The fresh tank is reached by snapping the large cardboard label off from the ends.

A very aggressive tape is then used to seal these holes. The metallized tape sometimes used for write protect tabs works fine. It is super important to get a permanant seal, for any toner dumped wholesale into your printer can cause permanant damage.

To refill your modified cartridge, remove the bottom tape and dump the spent toner out. Do this outside, preferably in a slight breeze. Tap the "notches" along the tank as you do this to release any caked toner. Reseal this hole after emptying.

Next, remove the fresh supply tank filling hole, and use a plastic funnel to dump a bottle of toner into the supply tank. Seal this hole. Add a label to your cartridge that keeps track of its history. Be sure to use the little green tool to clean the corona wire in the cartridge. Then run off fifty copies to make sure everything is working properly.

Finally, you also have to replace the wiper pad over on the fusion roller assembly. Carve the old white pad out of its black holder with a screwdriver and drop a new one in place. Secure it with white glue.

Yes, you can also refill the copier cartridges, but this takes a different toner using a slightly different technique. But, under no circumstances should you try to use laser printer toner in a copier or vice versa. The two toners are totally different.

The new SX cartridges can also be refilled, but there are problems at present. The drum here is much smaller and much less scratch resistant. Very often, the SX drum will scratch even before a first refill is even possible.

However; if you take a brand new SX cartridge and remove all of the factory toner from it, and then add a suitable drum lubricant and a third party refill toner, you can easily get several refills without any scratches.

Now, not even the most die hard of cynics would conclude that a company would purposely implant some abrasive into their toner to grind up their own drums, just to deprive their own end users of the highest possible image quality at the lowest cost.

On the other hand, it is painfully obvious that several rather simple changes in toner chemistry can be made that could dramatically reduce the SX drum scratching.

So, for right now, the SX cartridge per-page toner costs are substantially higher than CX toner costs. They also are likely to stay that way, at least for a while. More on this as the drama unfolds.

Tell me all About Perspective Transformations

As you found out last month, I've been having a lot of fun lately with all of my new perspective drawing routines. These can let a PostScript language printer and AppleWriter produce some absolutely stunning graphic images. Unlike all of the far more expensive routes to perspective drawing, my routines will automatically handle any true perspective *lettering* (!) of any style in any font, along with precise perspective circles

```
%  drawapple - draws a "street legal" apple. Enter with the currentpoint

%  set and apple size on stack


/apple1 {/Symbol findfont [1 0 0 1 0 0] makefont setfont} def

/apple1- {/Symbol findfont [-1 0 0 1 0 0] makefont setfont} def


/drawapple {gsave 1.32 mul dup scale gsave 0.04 setlinewidth 1 setlinecap

currentpoint .28 add translate 0 0 moveto 0 .05  -.05 .12  -.10 .16 rcurveto

stroke grestore currentpoint -.36 -.36 rmoveto .72 0 rlineto 0 .665 rlineto

-.18 0 rlineto -.18 -.06 rlineto -.18 +.06 rlineto -.18 0 rlineto closepath

clip 0.35 sub exch .41 sub exch moveto apple1 (\360) show currentpoint

exch 0.03 add exch moveto apple1- (\360) show grestore} def


% demo - remove before use . . .
```

Fig. 6 – PostScript code for a "street legal" apple.

```
133 25 {dup mul exch dup mul add 1.0 exch sub} setscreen 320 450

moveto 0.99 setgray 470 drawapple 0 setgray 475 175 moveto 100

drawapple gsave 150 80 translate 10{0 0 moveto 30 drawapple 40 0
```

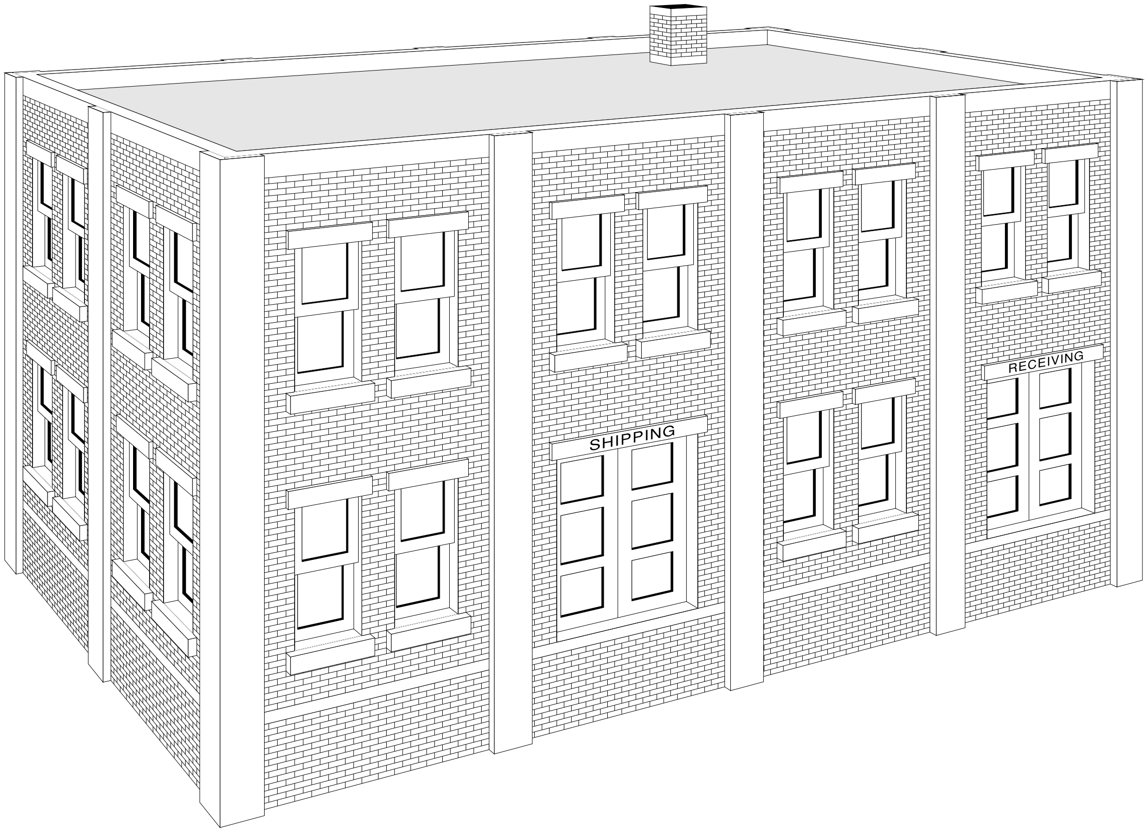


Fig. 7 – Another (yawn) AppleWriter text file.

and arcs. You can now even do an animated flyby, changing both your position and viewpoint with a very few keystrokes. Individual objects can also all be *seperately* rotated.

What I would like to do here is share with you all of the fundamental perspective transformations. Chances are you can apply all these to most any old graphics program on most any computer, even using plain old HIRES out of Applesloth. And it sure is fascinating to fly over a building with nothing but a couple of keystrokes. Only simple high school math is involved.

The perspective most people use most often is known as *two point* perspective in which all vertical lines stay that way. The simpler *one point* perspective that you would get while looking down a long hallway is just a special and centered case of this two point perspective.

At any rate, figures four and five show you both of the transformations as needed to convert a three dimensional object into a two dimensional representation on a sheet of paper or your computer's display screen.

Let's assume you had a giant plane of glass lying between you and your subject. We'll call this pane of glass our *picture plane*. We will start an origin at the center bottom of the picture plane. We'll use **xx** for the back and forth distance with right positive. We'll then make **yy** the distance into the picture, with the farther from you being more positive. And, we will use **zz** for up and down with positive up.

Next, we will assume you will be standing pretty far out in front of the picture plane. We can call **xo**, **yo**, and **zo** the distance from the plane to your eyeball. In general, for the two point perspective to work, you will want to stand fairly far back, using a rather large **yo**. Otherwise, your perspective illusion might distort and fail. Many times, you might want your **zo** to match a standard eyeball height, particularly on architectural drawings.

We'll call capital **X** the back and forth distance on both the final paper and your picture plane, and capital **Y** the up and down distance on the paper and the plane. As figures six and seven show us, two simple and similar traingles are all that is needed to transform your perspective image to the final page.

All that my routines or your new computer programs have to do is solve these similar triangles for each and every endpoint of each and every line to be drawn. While rather painful for a people, this is totally and utterly trivial for a computer.

Give me a call if you do need any more input on this.

What is this month's PostScript utility?

We will have a pair of them this month. The first, over in figure six, shows you how you might draw a "street legal" apple that a student of mine needed for some of the shipping crates in her orchard.

This is a good example of how you can modify existing characters and symbols to create new ones. In this case, we "unbit" the apple by printing it backwards on top of itself, and used a clipping boundary to get rid of the leaf. A new stem was then added by using a single cubic spline.

While this should be a totally and completely legal image, I would still hesitate to use it for any purpose that was even vaguely related to computers in any manner.

This reverse printing stunt is also useful to turn other images around, especially the pointing hands of the Zapf Dingbat font. Details are left as an excercise for the student.

One gotcha: The "bitten" apple is available as *Symbol* code \ 360 on the LaserWriters, but may not be present in non-Apple firmware. So much for device independence.

Figure seven is a little something I quickly threw together out of AppleWriter. This particular building was chosen because it gets fiendishly difficult to nearly impossible to draw when you use either *Illustrator* or *Cricket Draw*, yet it is almost trivial when done using a suprisingly short AppleWriter text file. Yes, this will translate to other systems.

Some of the key code involved is shown in figure eight. Write or call for a printed listing of all the core routines. You can also get this code ready-to-run in either my *PostScript Show and Tell* or my *PostScript Perspective Drawing* packages.

Note particularly that each and every brick is shown individually in its true perspective, that all of the lettering is also correctly shown in full and true perspective, and that the entire building can be rotated over a very wide range just by changing a few keystrokes.

Yes, as the building rotates, all of the "insets" and "outsets" will automatically take care of showing only those parts that will be seen correctly for that view.

And, yes, I do have lots more perspective images like this one. Also a few that include circles, arches, and even arcs in true perspective. Even some fireplace logs. Write or call if you need more info.

As a second contest that pretty near anyone can enter, just count the bricks used in the building. We'll have all the usual stuff as prizes.

Don't forget to put the cat out.

```
%  Copyright c 1988 by Don Lancaster & Synergetics, Box 809, Thatcher
%  AZ, 85552, (602) 428-4073. All rights reserved. Personal use
%  permitted so long as this header remains present and intact.

%  This illustration requires my two-point perspective utilities.

-90 rotate -792 0 translate  250 550 translate 0 setlinewidth /xo 00
def /yo 700 def /zo 400 def /objrot 25 def

/brickhi 2.5 def /brickwide 8 def /numbricksx 67.5 def /numbricksy 34.5
def /numbricksz 108 def

/bldz numbricksz brickhi mul def /bldx numbricksx brickwide mul def /bldy
numbricksy brickwide mul def

% main building outline
0 0 0 pm bldx bldy bldz pbox

% sunken roof
/insetbaseshade 0.9 def /insetwallshade 1 def /pptwide 10 def /ppthigh 10
def pptwide dup bldz prm bldx pptwide 2 mul sub bldy pptwide 2 mul sub
ppthigh xyinset

% add bricks to the walls
save /brickhold exch def 0 0 0 pm numbricksx numbricksz 4 sub xzbrickwall
0 0 0 pm numbricksy numbricksz 4 sub yzbrickwall clear brickhold restore

% the floor slab
0 0 0 pm 14 brickhi mul zrm bldx 2 brickhi mul xzrect 0 0 0 pm 14 brickhi
mul zrm bldy 2 brickhi mul yzrect

% columns
0 0 0 pm 16 brickwide mul 0.5 brickwide neg mul 0 prm 4 {2.5 brickwide mul
0.5 brickwide mul bldz pbox 16.5 brickwide mul xrm} repeat 0 0 0 pm 0.5
brickwide neg mul 16 brickwide mul 0 prm 2 {0.5 brickwide mul 2.5 brickwide
mul bldz pbox 16.5 brickwide mul yrm} repeat 0 0 0 pm 0.5 brickwide mul
neg dup 0 prm 2.5 brickwide mul bldz xzrect 0 0 0 pm 0.5 brickwide mul
neg dup 0 prm 2.5 brickwide mul bldz yzrect

% topwall
/aa {brickwide 0.5 mul} def /bb {brickwide 2.5 mul} def /cc {brickwide 14
mul} def newpath 0 0 0 pm gsave bldz zrm bldx xrd bldy yrd bldx neg xrd
closepath 1 setgray stroke grestore 0 0 0 pm aa neg aa neg bldz prm 4{bb
xrd aa yrd cc xrd aa neg yrd} repeat bb xrd 2{bb yrd aa neg xrd cc yrd aa
xrd} repeat bb yrd 4{bb neg xrd aa neg yrd cc neg xrd aa yrd} repeat bb neg
xrd 2{bb neg yrd aa xrd cc neg yrd aa neg xrd} repeat closepath stroke

% windows
4 25  10 25  4 68  10 68 20.5 68  26.5 68  37 68  43 68 37 25  43 25  53.5
68 59.5 68   11 {xzdhwindow} repeat  26.5 25 20.5 25 10 25 4 25 26.5 68
20.5 68 10 68 4 68 8 {yzdhwindow} repeat

% shipping door
20.5 16 xzshippingdoor 54 16 xzshippingdoor

% chimney
50 26 bldz ppthigh sub brickchimney

% some lettering
/Helvetica-Bold findfont [9.5 0 0 7 0 0] makefont setfont

/pixelproc {0 3.6 moveto 0.4 0 (SHIPPING) ashow} def /pixelprocwidth 50
def /pixelprocheight 10 def 0 0 0 pm 21.6 brickwide mul 0.2 41.2 brickhi
mul prm pixellineremap

/pixelproc {0 3.6 moveto 0.4 0 (RECEIVING) ashow} def /pixelprocwidth 70
def 0 0 0 pm 54.8 brickwide mul 0.2 41.2 brickhi mul  prm pixellineremap
showpage
```

Fig. 8 – PostScript Language code for the old building.

Don Lancaster's

ASK THE GURU

May, 1988

PostScript border builder
IIgs toolset dependencies
Getting started with APW
Transparent laser printing
Stock market price records

Jerry Cline and his great crew at the *AZ Apple* user group are now putting together a super whiz bang regional computing show and seminar. I will be there hosting a PostScript mini-course or two for you, while Bee will be showing and telling with lots of hands-on stuff for all of you beginning desktop publishers.

Other galaxy-famous dignitaries will include both Roger Wagener and Randy Hyde on machine language, and Greg Schaeffer on telecommunications, and possibly even several name brand Apple Computer employees of note.

You are specifically invited to attend. It will be at the Safari Hotel in Scottsdale, Arizona on June 11th and the 12th. Cost for the two-day show and for all the seminars and courses, is around $25. For more details, give Jim Ransom a call at (602) 276-9332.

In fact, virtually every Apple IIc, IIe and IIgs machine and assembly language programming author of note just may be there, since Randy Hyde is now coordinating that Western Design Center "Orca roast" for the very same weekend. The object here is to replace ORCA/M with a logical, useful, and non-quirky development system for the IIc, IIe, IIgs, and the upcoming -**censored**-. For more details on this, give Randy a call at (714) 735-7714.

All of them New Mexico orcas is really tough, unless you cook them for a long time over a very slow fire. Indian style.

these tools must already be installed

✔ = required
● = recommended

to use this tool	1. Tool locator ($01)	2. Memory manager($02)	3. Misc. Tools ($03)	4. QuickDraw II ($04)	5. Desk Manager ($05)	6. Event Manager ($06)	7. Scheduler ($07)	8. Sound Manager ($08)	9. Desktop Bus ($09)	10. SANE Toolset ($0A)	11. Integer Math ($0B)	12. Text Tools ($0C)	13.	14. Window Mgr. ($0E)	15. Menu Manager ($0F)	16. Control Manager ($10)	17. System Loader ($11)	18. Quickdraw Aux ($12)	19. Print Manager ($13)	20. Line Edit ($14)	21. Dialog Manager ($15)	22. Scrap Manager ($17)	23. Standard Files ($18)	24.	25. Note Synthesizer ($19)	26.	27. Font Manager ($1B)	28. List Manager ($1C)
1. Tool locator ($01)																												
2. Memory manager($02)	✔																											
3. Misc. Tools ($03)	✔	✔									✔																	
4. QuickDraw II ($04)	✔	✔	✔																									
5. Desk Manager ($05)	✔	✔	✔	✔										✔	✔	✔				✔	✔	✔						
6. Event Manager ($06)	✔	✔	✔	✔	✔				✔																			
7. Scheduler ($07)	✔	✔	✔																									
8. Sound Manager ($08)	✔	✔	✔																									
9. Desktop Bus ($09)	✔																											
10. SANE Toolset ($0A)	✔	✔																										
11. Integer Math ($0B)	✔																											
12. Text Tools ($0C)	✔																											
13.																												
14. Window Mgr. ($0E)	✔	✔	✔	✔		✔									●	✔												
15. Menu Manager ($0F)	✔	✔	✔	✔		✔								●		●												
16. Control Manager ($10)	✔	✔	✔	✔		✔								✔	●													
17. System Loader ($11)	✔	✔	✔																									
18. Quickdraw Aux ($12)	✔	✔	✔	✔																								
19. Print Manager ($13)	✔	✔	✔	✔		✔								✔	✔	✔		✔		✔	✔						✔	✔
20. Line Edit ($14)	✔	✔	✔	✔		✔																✔						
21. Dialog Manager ($15)	✔	✔	✔	✔		✔								✔	✔	✔				✔								
22. Scrap Manager ($17)	✔	✔																										
23. Standard Files ($18)	✔	✔	✔	✔		✔								✔	✔	✔				✔	✔							
24.																												
25. Note Synthesizer ($19)	✔	✔						✔																				
26.																												
27. Font Manager ($1B)	✔	✔	●	✔							●			●	●	●				●	●							●
28. List Manager ($1C)	✔	✔	✔	✔		✔								✔	●	✔												

Fig. 1 – Apple IIgs toolbox interdependencies.

Yes, a post-show *tinaja quest* or two could be arranged. Call me for more info on this.

Apple has a new video out. You can borrow a free copy from your local user group if you do faithfully promise to never ever return it.

Unlike most of Apple's previously outstanding videos, this one is right down there with *They Saved Hitler's Brain* and *Godzilla versus the Night Nurses*. The high point here is where you stare at a sign that says "New England Life" for half an hour. I'd guess speeches in Boston are all supposed to be dull. Oh well . . .

I'd say "nice try", only it isn't.

Apple has also introduced a new version of UNIX for the Mac called A/UX. The language is a mere 50 megabytes in length and easily fits on seventy 3-1/2 inch disk sides. The pocket reference card that goes with it costs $650, and is tersely crammed into fourteen oversize volumes of a mere 6000 total pages.

Apple has just announced a new CD/ROM drive, that is priced at $1200. Probably the first disks you might expect will be PostScript fonts from Adobe, the writer's tools from Microsoft, a few major library reference works including *Books in Print*, clip art for desktop publishing, some outstanding utilities supplied by the *BMUG* user group, and perhaps a motley collection or two of the worst and poorest of the burgeoning Hypercard stackware.

I am not sure just yet, but this should work on a IIe or IIgs having a SCSI interface card installed. Stay tuned for more details.

It will require some time before any genuinely useful CD/ROM disks become widely distributed at sane prices. A national phone directory would sure be nice, as would a complete history of all stock prices. What I am ultimately waiting for, though, are the USGS topo maps on disk. This one may take a while. So will the printer. Sigh.

Expect some sort of a super-secret IIgs upgrade "real soon now" with a higher clock speed and possibly an in- creased vertical screen resolution.

I still haven’t yet received my new LaserWriter NTX. But, based upon some quickie tests on a dealer demo, I do have some predictions on the ultimate outcome: First and foremost, this is a great machine that, except for some serious problems, just might set a new standard for low end laser printers for several years to come.

The apparent new LaserWriter II problems are these:

(1) Per page toner costs can be as much as *fifteen times* higher than on the older LaserWriters, and all the images are not all that much blacker than those you get from the latest of third-party toner refills for the older CX machines;

(2) You still will have to go to *Hewlett-Packard* for manuals, for parts and for service info. Apple appears to be paying H-P a $50 rebate for each LaserWriter sold, since the machine is much more useful when you have the outstanding H-P manual #33440-90904 on hand.

(3) The LocalTalk network will horrendously slow the NTX "click to clunk" time for many jobs down so badly that the NT *without* LocalTalk is often ridiculously faster than the NTX *with* LocalTalk. This is especially a problem on the IIgs;

(4) The suggested list prices are an outright joke that nobody could possibly treat seriously; and finally . . .

(5) The hard disk usage and its support is both very poorly done and excessively restrictive. At the very least, the disk should double as a network server, and as a job printer that is possibly even as powerful as AppleWriter’s WPL. A "snapshot" ability for final bitmaps would be even handier.

Meanwhile, NEC has come out with a new LC-790 PostScript "laserless" laser printer using a LED strip array that seems to be garnering an awful lot of user helpline support. Eveyone who has gotten even remotely near this machine has been wildly enthuasastic. And, yes, those NEC toner cartridges are both refillable and reconditionable. Either factory or third-party.

Actually, those very high LaserWriter II per-page toner costs could kill the new machines in their tracks. A knowledgeable LaserWriter Plus owner can do his own toner refills for as little as 0.32 cents per page. The NEC machine toner costs around 1.2 cents per page, and this can be cut in half or less by the usual third-party techniques. This compares with a nickel per page "their way" toner

```
       ;  CS.DEMO.1  An APW example of how to start up various
       ;             toolsets, get quickdraw in gear, clear the
       ;             640 super HIRES screen to a selected color
       ;             and hang the machine for viewing.

              KEEP   CSD.1        : save final copies
       MAIN   START               ;
              CASE   ON           ; be case sensitive

              MLOAD /libraries/ainclude/m16.locator
              MLOAD /libraries/ainclude/m16.memory
              MLOAD /libraries/ainclude/m16.misctool
              MLOAD /libraries/ainclude/m16.event

              _TLStartUP           ; start the tool locator
              pla                  ;
              sta    MYID          ; and save the user i.d.

              _MMStartUp           ; start the memory manager
              _MTStartUp           ; start the misc. tools

              lda    #$23          ; begin memory manager
              pha                  ;
              lda    #$00          ; initializes event
              pha                  ;
              lda    #$00          ; queue
              pha                  ;
              lda    #$00          ; mouseclamp
              pha                  ;
              lda    #$00          ;
              pha                  ;
              lda    #$00          ;
              pha                  ;
              lda    MYID          ; user i.d. again
              pha                  ;
              _EMStartup           ;

              MDROP /libraries/ainclude/m16.locator
              MLOAD /libraries/ainclude/m16.quickdraw

              lda    #$20          ; start quickdraw
              pha                  ;
              lda    #$80          ; master SCB
              pha                  ;
              lda    #$00          ; use full screen
              pha                  ;
              lda    MYID          ; and current i.d.
              pha                  ;
              _QDStartUp           ; and make the call

              lda    #$00          ; set the color table to zero
              pha                  ;
              pea    ctab0|-16     ; high address of color table
              pea    ctab0         ; low address of color table
              _SetColorTable       ;

              lda    #$80          ; set screen control bytes
              pha                  ;
              _SetAllSCBs          ; for all 640 resolution

              -GrafOn              ; turn on super HIRES screen

              lda    #$5555        ; clear the screen to red
              pha                  ;
              _ClearScreen         ;

       Hang   JML    Hang          ; hang for viewing

       MYID   DC    I’0’           ; user ID goes in here

       ctab0  dc    I2’$0000,$000f,$0ff0,$0fff’ ;standard color table
              dc    I2’$0000,$0f00,$00f0,$0fff’ ;
              dc    I2’$0000,$000f,$0ff0,$0fff’ ;
              dc    I2’$0000,$0f00,$00f0,$0fff’ ;

              END
```

Fig. 2 – APW example shows how to start up toolsets.

costs for the LaserWriter II. Which is over a 15:1 cost penalty.

It is absolutely, totally, and utterly inexcusable to not offer any refillable LaserWriter SX cartridges as factory or third-party options. Needless to say, stay tuned for more on all this.

Steve Ciarcia has started up a new hacker's magazine known as *Circuit Cellar Ink* that does look like a real winner. And, as a reminder, if you are into electronic hardware at all, be sure to check into my *Hardware Hacker* sister column to this one that is going great guns over in *Radio Electronics* magazine.

Let's see. Which advetorial should I get you to ignore this month? How about my new *PostScript* language perspective drawing package, and my *PostScript Show & Tell*, or my complete set of *Ask the Guru* reprints? Write or call if you need more info on these or any others.

The big stupendous breakthrough deal this month is an automatic and insanely intelligent PostScript border machine. But first . . .

What is the IIgs programmer's dilemma?

Here is something that I've been kicking around with my students and with some user groups. I would also like some input from you on this.

The IIgs may have thrown out the baby with the washwater.

On the Apple II and II+, any old seventh grader could hack around long enough till they got a something that actually seemed to work. He could jot it down on a napkin and share it with others. This would go on building and bootstraping his way along, until he was a millionare CEO head of his own software firm.

The IIgs did offer us "new" and "more powerful" ways of doing some things. Included was an incredibly complex toolbox and the ability to use "icons". We can now do the "big machine" things that we were never able to do before. With outstanding sound and superb graphics.

But only at a heinous price. First, it seems that the IIgs usually ends up much *slower* than the IIe, because of the layer-upon-layer "onion effect" of all the complex relocatable software involved. We have such absurdities as taking 130 microseconds in the

```
%  Copyright c 1988 by Don Lancaster & Synergetics, Box 809, Thatcher AZ, 85552,
%  (602) 428-4073. All rights reserved. Personal, non-commercial use permitted so long as
%  this header remains both present and intact. Work in Progress disk costs $39.50.

%  predefine the following, with or without a dictionary: mainbproc, cornerbproc, auxbproc,
%  maindeltax maindeltay auxdeltax, auxdeltay, cornerdeltax, cornerdeltay, breductionfactor

% enter with -bordleft- -bordwidth- -bordtop- -bordheight- -bordthick- drawsymmetricborder

/drawsymmetricborder { /approxscale exch def /bordheight exch def /bordtop exch def
/bordwidth exch def /bordleft exch def /bordwidth bordwidth breductionfactor 2 mul approxscale
mul sub def /bordheight bordheight breductionfactor 2 mul approxscale mul sub def /bordtop
bordtop breductionfactor approxscale mul sub def /bordleft bordleft breductionfactor
approxscale mul add def

% this calculates the approximate width needed
bordwidth approxscale div cornerdeltax sub cornerdeltay sub maindeltax sub maindeltax
auxdeltax add div /trialfitx exch def

% this adjusts the scale for an almost exact fit
trialfitx ceiling cvi /numx exch def trialfitx dup ceiling div approxscale mul /exactxscale
exch def

% this adjusts the xscale for a main more exacter fit
bordwidth cornerdeltax cornerdeltay add maindeltax numx 1 add mul add auxdeltax numx
mul add exactxscale mul div exactxscale mul /exactxscale exch def

% this calculates the approximate height needed bordheight
approxscale div cornerdeltay sub cornerdeltax sub maindeltax sub maindeltay auxdeltay add
div /trialfity exch def

% this adjusts the scale for an almost exact fit
trialfity ceiling cvi /numy exch def trialfity dup ceiling div approxscale mul /exactyscale
exch def

% this adjusts the yscale for an main more exacter fit
bordheight cornerdeltay cornerdeltax add maindeltay numy 1 add mul add auxdeltay numy
mul add exactyscale mul div exactyscale mul /exactyscale exch def

% do the mainproc and auxproc all the way around
gsave bordleft cornerdeltax maindeltax 2 div add exactxscale mul add bordtop translate gsave
numx 1 add {0 0 gsave exactxscale exactyscale scale mainbproc grestore maindeltax
auxdeltax add exactxscale mul 0 translate} repeat grestore maindeltax 2 div auxdeltax 2 div
add exactxscale mul 0 translate gsave numx {0 0 gsave exactxscale exactyscale scale
auxbproc
grestore maindeltax auxdeltax add exactxscale mul 0 translate} repeat
grestore grestore % top

gsave bordleft bordwidth add bordtop cornerdeltay maindeltay 2 div add exactyscale mul
sub translate gsave numy 1 add {0 0 gsave exactyscale exactxscale scale -90 rotate mainbproc
grestore 0 maindeltay auxdeltay add exactyscale mul neg translate} repeat grestore 0
maindeltay 2 div auxdeltay 2 div add neg exactyscale mul translate gsave numy {0 0 gsave
exactyscale exactxscale scale -90 rotate auxbproc grestore 0 maindeltay auxdeltay add
exactyscale mul neg translate} repeat grestore grestore % right

gsave bordleft cornerdeltax maindeltax 2 div add exactxscale mul add bordtop bordheight sub
translate gsave numx 1 add {0 0 gsave exactxscale exactyscale scale 180 rotate mainbproc
grestore maindeltax auxdeltax add exactxscale mul 0 translate} repeat grestore maindeltax 2
div auxdeltax 2 div add exactxscale mul 0 translate gsave numx {0 0 gsave exactxscale
exactyscale scale 180 rotate auxbproc grestore maindeltax auxdeltax add exactxscale mul
0 translate} repeat grestore grestore % bottom

gsave bordleft bordtop cornerdeltay maindeltay 2 div add exactyscale mul sub translate gsave
numy 1 add {0 0 gsave exactyscale exactxscale scale 90 rotate mainbproc grestore
0 maindeltay auxdeltay add exactyscale mul neg translate} repeat grestore 0 maindeltay 2
div auxdeltay 2 div add neg exactyscale mul translate gsave numy {0 0 gsave exactyscale
exactxscale scale 90 rotate auxbproc grestore 0 maindeltay auxdeltay add exactyscale mul
neg translate} repeat grestore grestore % left

% and the corners all go on last ...
gsave bordleft bordtop translate  exactxscale exactyscale scale 0 0 cornerbproc
grestore % upper left

gsave bordleft bordwidth add bordtop translate exactxscale exactyscale scale -90 rotate
0 0 cornerbproc grestore % upper right

gsave bordleft bordwidth add bordtop bordheight sub translate exactxscale exactyscale
scale 180 rotate 0 0 cornerbproc grestore % lower right

gsave bordleft bordtop bordheight sub translate exactxscale exactyscale scale 90 rotate 0 0
cornerbproc grestore % lower left

%  report squashticity to host
(  The current squashticity factor is ) print exactyscale exactxscale div == flush} def
```

Fig. 3 – The bountifully bodacious better border builder.

fast mode on a IIgs to find out if a port character is available to be read – compared to only 6 on the IIe. And none of the new gee-whiz IIgs word processors can even think of holding a candle to good old AppleWriter on a IIe, when it comes to no-nonsense operating speed and raw power.

But worst of all is the "us versus them" effect. To use the IIgs "their way", you first have to run out and buy several hundred dollars worth of APW Apple Programmer’s Workshop software and those thirteen volumes needed to support it. And you have to spend well over a year in time and an incredible amount of discipline and frustration to be able to master the APW environment.

Sadly, your final results cannot be passed on to just anyone. You can only talk to people who know, use, and understand APW. All seven of them. There are no machine language listings for any relocatable "big machine" code. Only a highly complex file structure is present. A structure that is quite hard to either list or understand, unless worked with from inside APW itself.

What in effect has happened is that a IIgs fence has been built, with users on one side, and the programmers on the other. And that fence is very hard to climb. Impossible for some.

This, of course, is the exact antithesis of what made the Apple II and II+ so great – anyone could stumble into anything. And they did. That is precisely what made the Apple such a stupendously innovative product in the first place.

Now, I will freely admit that the IIgs "us versus them" fence is not nearly as horrendous as the one on the Mac. On the Mac, their fence is now electrified, strafed, and continuously patrolled by AIDS-infected pit bulls. Mean mothers.

It appears that far and away the best IIgs code will be written the "old way" with custom and hand-crafted fixed position, listable, code modules that either will completely ignore the toolbox, or else use only the genuinely useful parts of acceptable speeds.

The centralmost fatal flaw in the IIgs is the monumental amount of frustration that is involved in serious program development. I have seen rank beginners generate absolutely outstanding machine language code on the II+ or IIe. On the IIgs, two of my best students ever were unable to produce a green hexagon on the IIgs screen in an entire semester’s work.

Very simply, APW sucks and has to go if the IIgs is to survive. We absolutely must have some new, very simple and highly intuitive ways for beginners to generate code that is both useful and rewarding.

Are you an us or a them? Has Apple gilded the goose, and killed it in the process? Please let me know all your thoughts on this. The world awaits your comments.

Tell me About the IIgs Toolset Interdependencies

The key to all of the "big machine" capabilities of the IIgs are the various tools as contained in the three dozen or so toolkits. Several of these are ROM based, while others are loaded off disk only when and as needed.

A tool is used by making a *toolbox call*. Basically what you do is shove some stuff onto your stack, load the wide X register with the tool number and the toolkit number, and then do a long subroutine call by asking for a *JSL E1 0000.* The tool then does its thing, often changing the graphics and sound states, and usually returning a bunch of goodies to you via the stack.

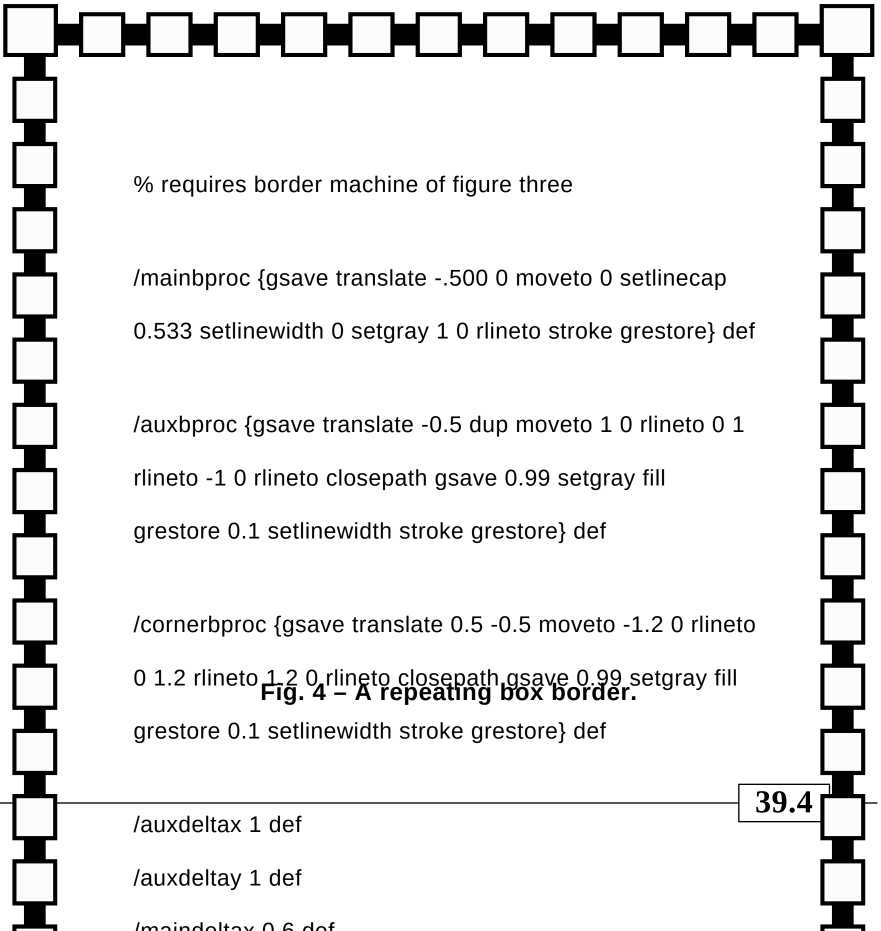
The stuff shoved onto or off of the stack can be values, data, or pointers to tables or needed address locations. The rules are different for each and every of the hundreds of tools. The full details appear in the two volume *Apple IIgs Toolbox Reference Manuals*, which is now at long last available in its final form from APDA.

APW does include some powerful macros that can simplify making a toolbox call. But you still will have to carefully set up the stack before making this call, following the rules for that tool.

What gets real messy fast is that some tools need other tools already present or they will not work. These are called *toolbox interdependencies*, and many of them are listed in figure one. Before a tool or toolbox can be used, all of the precursor tools must already be in place and ready to use.

Thus, you have to be very careful about when and how you select a tool or a toobox, or some highly ungood things will happen.

Very interestingly, tool 14 requires tool 16 to be installed first. And – you guessed it – tool 16 requires tool



```
% requires border machine of figure three

/mainbproc {gsave translate -.500 0 moveto 0 setlinecap
0.533 setlinewidth 0 setgray 1 0 rlineto stroke grestore} def

/auxbproc {gsave translate -0.5 dup moveto 1 0 rlineto 0 1
rlineto -1 0 rlineto closepath gsave 0.99 setgray fill
grestore 0.1 setlinewidth stroke grestore} def

/cornerbproc {gsave translate 0.5 -0.5 moveto -1.2 0 rlineto
0 1.2 rlineto 1.2 0 rlineto closepath gsave 0.99 setgray fill
grestore 0.1 setlinewidth stroke grestore} def

/auxdeltax 1 def
/auxdeltay 1 def
/maindeltax 0.6 def
```

Fig. 4 – A repeating box border.

14 to be installed first. The rule is this: The window (14), control (16) and menu manager (15) should be considered as one unit and should be started in the order just shown.

These dependencies are apparently a "moving target", since newer firmware versions and changes in APW are continually being made.

Show me a Simple APW And Toolset Example

You've got to be kidding. There are no simple APW toolset examples.

At any rate, figure two shows you how to use the APW macros to start up a bunch of the various toolsets, get quickdraw in gear, next clear the screen to a favorite color, and then hang up the machine so that you can view your handiwork.

This can be the starting point for a program that really does something useful. Hint: the next step is to open some windows.

I am personally getting into APW in a big sort of way, so let me know what you want to see in the way of additional toolbox secrets.

Once again APDA, short for the *Apple Programmers and Developer's Association*, is where you go to get additional tech info on most Apple and third party products. Note that this is a seperate and independent organization that anyone can join, and is not related in any way to the Apple registered developer and the certified developer programs.

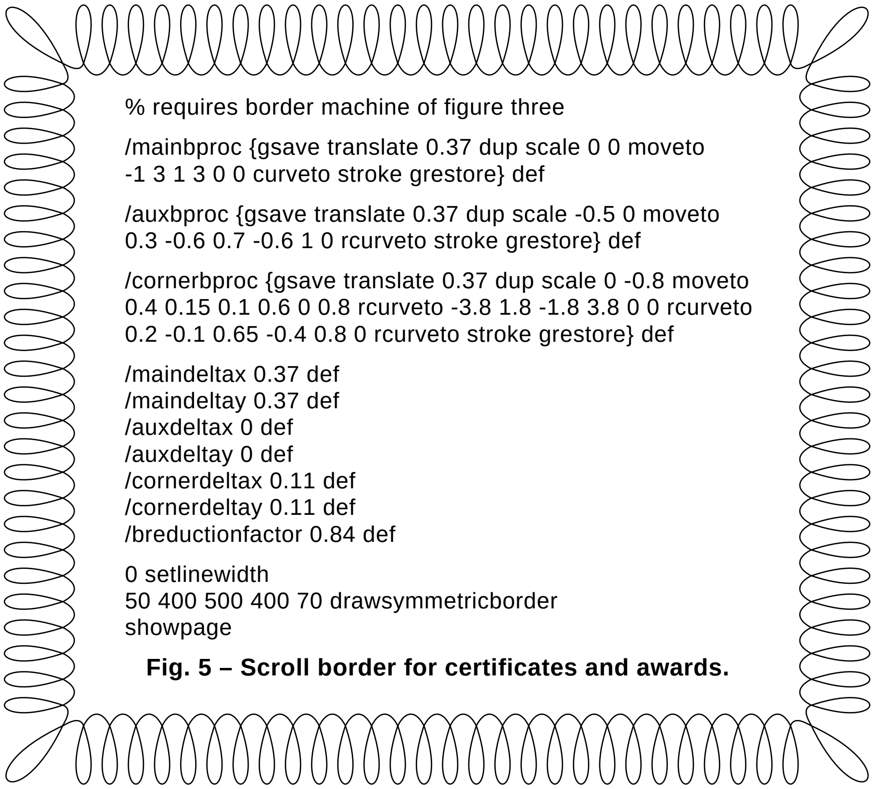
Where can I Get Stock Price Histories?

Obviously from the daily quotes in *The Wall Street Journal* or else the weekly ones in *Barrons*. In fact, out of the 387 plus magazines and other stuff that I subscribe to, the Wall Street Journal is my third favorite. Right after the *Whole Earth Review*, and, of course, good old numero uno *MAD* magazine.

Stock histories are also available on *Compuserve*, as well as through all the various software packages offered by leading brokers.

But far and away my most useful sources of stock price histories are those super cheap, ultra scungy, and downright obscure *Quote NY*, *Quote American*, and *Quote OTC*. These are available from *Harry Lankford* for around $7 or so each.

About the only complaint I have on *Quote OTC* is that it never picked up on *Adobe* stock. As we've seen in previous columns, nearly any idiot could have quintupled his money on this in the last two years, while a knowledgeable (and admittedly very lucky) investor could have gotten a 3000 percent return.



```
% requires border machine of figure three

/mainbproc {gsave translate 0.37 dup scale 0 0 moveto
-1 3 1 3 0 0 curveto stroke grestore} def

/auxbproc {gsave translate 0.37 dup scale -0.5 0 moveto
0.3 -0.6 0.7 -0.6 1 0 rcurveto stroke grestore} def

/cornerbproc {gsave translate 0.37 dup scale 0 -0.8 moveto
0.4 0.15 0.1 0.6 0 0.8 rcurveto -3.8 1.8 -1.8 3.8 0 0 rcurveto
0.2 -0.1 0.65 -0.4 0.8 0 rcurveto stroke grestore} def

/maindeltax 0.37 def
/maindeltay 0.37 def
/auxdeltax 0 def
/auxdeltay 0 def
/cornerdeltax 0.11 def
/cornerdeltay 0.11 def
/breductionfactor 0.84 def

0 setlinewidth
50 400 500 400 70 drawsymmetricborder
showpage
```

Fig. 5 – Scroll border for certificates and awards.

My current recommendation on Adobe is to buy at 24 and sell at 37. At this writing, several of the OTC stocks that have not recovered from last October's lateral arabesque do include *Circle Express*, *Carver*, and *Siliconix*. Check them out.

Just be sure to remember that this is free advice, and worth every cent of it. After all, look at me. I started out with nothing, and I still have most of it left.

Can you Laser Print Onto Transparent Materials?

You might want to laser print onto transparent material for an overhead transparency, for a printed circuit or dialplate art master, for any "repair" stick-on patches, for a store or auto window decal, or simply to get a high gloss final result.

There are two popular transparent graphics art materials, acetate and polyester. Polyester sometimes gets called by its DuPont "mylar" trade name. These are the same.

Available thicknesses vary all over the lot, but 2 mils (for self-stick) or 5 mils (for stand-alone) are often a good choice. Both clear and matte finishes are available

In general, toner sticks beautifully to either material, giving very dense and very solid blacks.

Acetate is cheap but has very poor dimensional stability and melts well below the fusion roller temperature. You should *never* feed unsupported acetate to a laser printer as it will certainly glop up the works. Some self-stick acetates can be run through a laser printer if the support backing is thick enough and stable enough. Ex- treme caution is required here.

Polyester does cost more, but it has excellent dimensional stability. It will safely withstand the fusion roller temperatures, even when it isn't properly supported. Thus, the polyester films will almost always be your far better choice.

A few sources. Those *James River* people now have free sample packets of their new *ProTech* polyester overhead transparency material available for you. *Xerox* has the same stuff at insane prices, but this is sometimes offered as a loss leader by *Quill.*

That great *Paper Plus* chain of discount paper stores also has a wide variety of transparent laser materials available. Be sure to check them out.

Dick Blick does carry both clear acetate and polyester in both regular and in self-stick styles. Be sure to ask for their sign painting and silk screen catalog, as well as for their regular art supplies catalog.

Pretty near any old blueprint or drafting materials supply house will stock polyester in both regular and self-stick, but not at reasonable prices. *Stanpat* is one major brand name; some others advertise in the *Design News* and the *Plan and Print* trade journals.

Finally, there's a converter house called *Catalina Plastics Inc.* that will custom cut you most any size and thickness of most any plastic with or without permanent or removable self-stick adhesive at very low prices – provided you are interested in a bare minimum of 5000 sheets or so. Their pricing is usually one-third to one-fifth of any retail source.

Here are two tips involving transparent materials: If you place a sheet of one mil mylar in contact with a paper and toner image, and then run it back through the printer while hand feeding a blank copy, that toner image will *Bakerize*, and convert the toner into a much more durable, blacker, and glossy image. This is particularly useful for business cards, but gets overwhelming in any larger text blocks.

It is utterly trivial to print backwards on a PostScript printer. Just use the *612 0 translate -1 1 scale* command. This lets you put all your self-stick transparent decals on the *inside* of a window or a glass door, where they will be considerably more weather and vandal resistant.

By the way, I would very much like to find a transparent material that has all of the following "magic" properties: It would be dimensionally stable at high temperatures. It would also reliably, but only "loosely" hold toner. What you would do with it is copy a 1:1 printed circuit pattern onto it, and then use this to transfer the toner directly to a printed circuit board for etching. The toner would serve as the actual etch resist.

This would greatly simplify the printed circuit production process, especially for students and hackers. So far, some of the materials I have tried have failed miserably, while there have been a few near misses.

A properly fused toner image does make an excellent etch resist. The problem is to get the toner from the laser printer drum onto the solid printed circuit board material reliably and without any distortion.

Let me know if you have a good solution to this, for there is a very immediate and very large market for the direct toner-based printed circuit etching. Particularly for prototypes and short runs. Virtually all printed circuit production products to date are either "high end" or useless trash. This one would solve a lot of big problems for a lot of people.

What is This Month's PostScript Utility?

Just an automatic and intelligent border machine, that's all. Unlike any other way of doing borders, you simply tell the routine where you want the border and how high and how wide to make it. You also tell it approximately how *thick* you want the border to be. The routine does the rest, automatically figuring out how many "lumps" are needed vertically and horizontally and then automatically giving you perfect corners and alignment for each and every lump.

Figure three does show you all the PostScript code involved. You first define three PostScript procedures for a *mainproc*, an *auxproc*, and for a *cornerproc*, along with their sizes. The only major restriction is that *mainproc* is supposed to be exactly "one" unit wide.

The border elements are laid down in an unusual sequence that simplifies your artwork tremendously. First, the *auxproc* elements go down. These are overwritten by the *mainproc* elements, which in turn will get themselves overwritten by the four *cornerproc* elements.

Figures four and five give you two ready-to-run examples for some typical borders. The scroll border shown is particularly useful for any certificates and awards. After you have defined a few borderprocs of your own, you might want to place them into a PostScript dictionary, where they can called out by a single literal name.

Some intelligent adjustments have to be made internally by the magic border machine. On some borders, particularly those with only a few elements on each side, the borderprocs might end up slightly squashed or stretched out. The border machine reports a *squashticity* factor back to the host. The optimum squashticity is 1.0. In rare instances, it might be necessary to slightly adjust the overall border height or width in order to get an acceptable squashticity.

Squashticity problems are utterly negligible for most popular border designs, especially if there are lots of elements on each side. The squashticity factor is strictly a "fine tuning" tool for perfectionists.

This border routine is intended for *symmetric* borders that "rotate" as you go around them. Another related routine is available for *plain* borders whose elements all point in the same direction. For instance, valentine hearts look kinda stupid when shown sideways or upside down; These will demand a plain border.

For this month's contest, just send me a new border that works well with my stupendous new border routines. There will be all my usual *Incredible Secret Money Machine* book prizes, along with an all expense paid (FOB Thatcher, AZ.) *tinaja quest* for two for the best border of all.

If you don't yet have access to a PostScript speaking laser printer, just send along any old sketch of any symmetric border that you would like to see automatically adjust itself to fit any image size and do so with perfect corner joints and perfect spacing.

Yes, I am now putting together a new PostScript language super border machine package for use with most personal computers. The goal is to have more than a thousand nontrivial borders ready-to-go as well as being extendable any manner you like, however fancy.

Do call or write me if you are at all interested in this or most anything else, PostScript, or whatever.

Don Lancaster's

ASK THE GURU

June, 1988

The Numeric IIgs tool list
Using padding compound
PostScript printed circuits
IIgs old disk drive adaptor
Selecting your laser printer

The FCC has dropped their plan for their proposed $6 per hour hacker telecomm tax, thanks in no small part to the cage rattling done by all of you *Computer Shopper* readers.

And, the foes of computer backup DAT digital audio tape have been dealt two severe blows. *NBS* first laughed them clear out of congress when they showed that their innane hardware DAT protection scheme just plain does not work, destroys the program integrity, and is trivially easy to cheaply bypass.

Meanwhile, DAT champion *Sony* put their money where their mouth was and bought *CBS* records outright. In the process, they told all the anti-DATs left over at CBS to sit down and shut up.

Naturally, all the anti-DATs have vowed to continue cutting off their nose to spite their face. After all, look at how those VCR's completely stopped the entire movie industry dead in its tracks. Uh huh.

Apple IIgs sales have been going through the roof and breaking all sorts of records. Apple has responded to this deplorable and despicable situation in the predictable manner – by slashing the price of the Mac so it now lists for less than that of a loaded IIgs system.

Recently, a secret survey revealed that Apple had erred by a factor of four on their estimate of the average age of a IIgs buyer. Their planned "free skateboard with each IIgs purchased" promo has been cancelled.

Unfortunately, the thinking behind it does still remain in place. Bubble gum cards, anyone?

Actually, you might expect across-the-board price cuts this summer on the IIgs and most everything Apple.

The Apple "suggested list prices" are totally meaningless anyway so long as you do shop around for a reasonable dealer. You will get the worst prices from the larger yuppie chains that include *Computerland* or *Businessland*, or from a dealer that has a "lock" on a regional market. The trick is to ask around at your local user group to find out who the good guys and the bad guys are in your particular area.

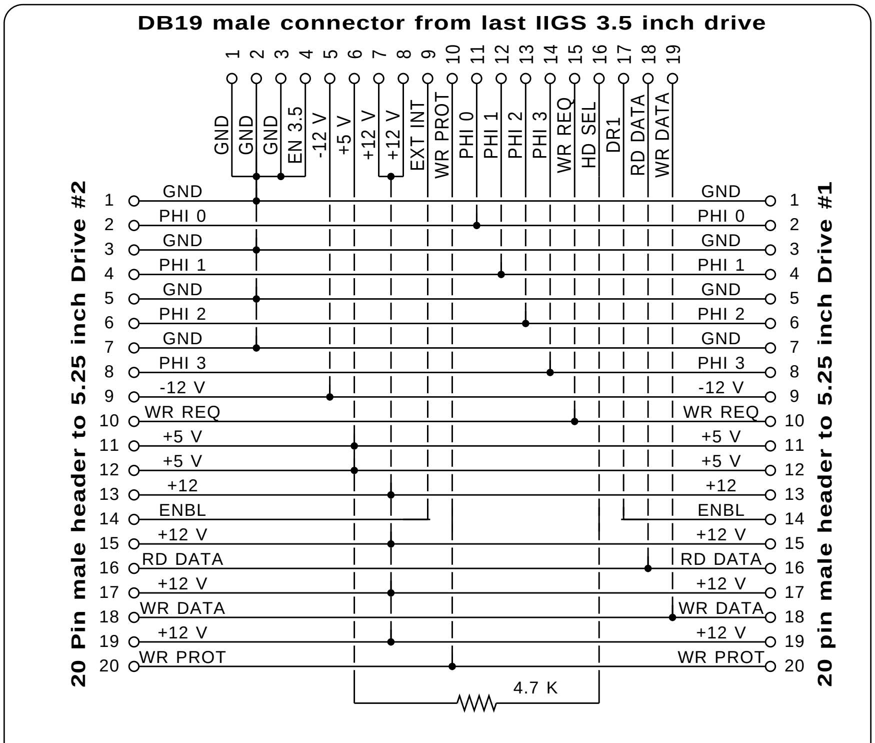


Fig. 1 - Apple IIgs old disk drive adaptor.

You can get a list of all your local Apple user groups by calling (800) 538-9696, Extension 500.

If you are at all involved with any sort of a school, be sure to check into some of Apple's great educational discounts. There's also a very lively used Apple market. I have been particularly impressed with the many offerings from *Shreve Systems*.

Let's see. I just got word on a low cost replacement for the Kroy Kolor machine that is based on the widely discounted *Canon* fuser unit. More details in a future column.

Adobe Systems has finally released the "green book". Which is volume three in their PostScript programming series. Yes, I do have copies on hand here if you cannot find them locally. This volume is on PostScript program design and has plenty of hands-on detailed examples in it.

And, yes, my new LaserWriter NTX finally showed up. But not in time for a thorough test and review this month.

Some quick answers . . .

Yes, this is an absolutely outstanding machine that should define what laser printing should and will be for the next several years.

Yes, each of those predicted major problems we looked at last month exist and are for real.

Yes, there is at least a 2:1 speedup, provided you do not let AppleTalk or your application program get in the way. In some cases, the speedup can and will end up negligible.

Yes, the NTX works beautifully with the IBM clones, and every other host computer under the sun. Various hand-shaking options are now both switch and software selectable.

No, the AppleTalk network is most definitely NOT needed to use this great machine. In fact, it often will

```
--------------------------------- TOOL #01 - TOOL LOCATOR ----------------------------
$0101- TLBootInit      $0601- TLStatus        $0D01- SetWAP          $1201- TLTextMountVolume
$0201- TLStartup       $0901- GetTSPtr        $0E01- LoadTools       $1301- SaveTextState
$0301- TLShutDown      $0A01- SetTSPtr        $0F01- LoadOneTool     $1401- RestoreTextState
$0401- TLVersion       $0B01- GetFuncPtr      $1001- UnloadOneTool   $1501- MessageCenter
$0501- TLReset         $0C01- GetWAP          $1101- TLMountVolume
------------------------------ TOOL #02 - MEMORY MANAGER -----------------------------
$0102- MMBootInit      $0B02- RestoreHandle   $1B02- FreeMem         $2302- HunlockAll
$0202- MMStartUp       $1002- DisposeHandle   $1C02- MaxBlock        $2402- SetPurge
$0302- MMShutDown      $1102- DisposeAll      $1D02- TotalMem        $2502- SetPurgeAll
$0402- MMVersion       $1202- PurgeHandle     $1E02- CheckHandle     $2802- PtrToHand
$0502- MMReset         $1302- PurgeAll        $1F02- CompactMem      $2902- HandToPtr
$0602- MMStatus        $1802- GetHandleSize   $2002- HLock           $2A02- HandToHand
$0902- NewHandle       $1902- SetHandleSize   $2102- HLockAll        $2B02- BlockMove
$0A02- ReallocHandle   $1A02- FindHandle      $2202- Hunlock
--------------------------- TOOL #03 - MISCELLANEOUS TOOLS --------------------------
$0103- MTBootInit      $0E03- WriteTimeHex    $1903- SetMouse        $2403- FWEntry
$0203- MTStartUp       $0F03- ReadAsciiTime   $1A03- HomeMouse       $2503- GetTick
$0303- MTShutDown      $1003- SetVector       $1B03- ClearMouse      $2603- PackBytes
$0403- MTVersion       $1103- GetVector       $1C03- ClampMouse      $2703- UnPackBytes
$0503- MTReset         $1203- SetHeartBeat    $1D03- GetMouseClamp   $2803- Munger
$0603- MTStatus        $1303- DelHeartBeat    $1E03- PosMouse        $2903- GetIRQEnable
$0903- WriteBBRam      $1403- ClearHeartBeat  $1F03- ServeMouse      $2A03- SetAbsClamp
$0A03- ReadBRam        $1503- SysFailMgr      $2003- GetNewID        $2B03- GetAbsClamp
$0B03- WriteBBParam    $1603- GetAddr         $2103- DeleteID        $2C03- SysBeep
$0C03- ReadBParam      $1703- ReadMouse       $2203- StatusID
$0D03- ReadTimeHex     $1803- InitMouse       $2303- InitSource
---------------------------------- TOOL #04 - QUICKDRAW ------------------------------
$0104- QDBootInit      $3804- SetSolidBackPat $6D04- OpenRgn         $A504- DrawString
$0204- QDStartUp       $3904- SolidPattern    $6E04- CloseRgn        $A604- DrawCString
$0304- QDShutDown      $3A04- MoveTo          $6F04- OffsetRgn       $A704- DrawText
$0404- QDVersion       $3B04- Move            $7004- InsetRgn        $A804- CharWidth
$0504- QDReset         $3C04- LineTo          $7104- SectRgn         $A904- StringWidth
$0604- QDStatus        $3D04- Line            $7204- UnionRgn        $AA04- CStringWidth
$0804- AddPt           $3E04- SetPicSave      $7304- DiffRgn         $AB04- TextWidth
$0904- GetAddress      $3F04- GetPicSave      $7404- XorRgn          $AC04- CharBounds
$0A04- GrafOn          $4004- SetRgnSave      $7504- PtInRgn         $AD04- StringBounds
$0B04- GrafOff         $4104- GetRgnSave      $7604- RectInRgn       $AE04- CStringBounds
$0C04- GetStandardSCB  $4204- SetPolySave     $7704- EqualRgn        $AF04- TextBounds
$0D04- InitColorTable  $4304- GetPolySave     $7804- EmptyRgn        $B004- SetArcRot
$0E04- SetColorTable   $4404- SetGrafProcs    $7904- FrameRgn        $B104- GetArcRot
$0F04- GetColorTable   $4504- GetGrafProcs    $7A04- PaintRgn        $B204- SetSysFont
$1004- SetColorEntry   $4604- SetUserField    $7B04- EraseRgn        $B304- GetSysFont
$1104- GetColorEntry   $4704- GetUserField    $7C04- InvertRgn       $B404- SetVisRgn
$1204- SetSCB          $4804- SetSysField     $7D04- FillRgn         $B504- GetVisRgn
$1304- GetSCB          $4904- GetSysField     $7E04- ScrollRect      $B604- SetIntUse
$1404- SetAllSCBs      $4A04- SetRect         $7F04- PaintPixels     $B704- OpenPicture
$1504- ClearScreen     $4B04- OffsetRect      $8104- SubPt           $B804- PicComment
$1604- SetMasterSCB    $4C04- InsetRect       $8204- SetPt           $B904- ClosePicture
$1704- GetMasterSCB    $4D04- SectRect        $8304- EqualPt         $BA04- DrawPicture
$1804- OpenPort        $4E04- UnionRect       $8404- LocalToGlobal   $BB04- KillPicture
$1904- InitPort        $4F04- PtInRect        $8504- GlobalToLocal   $BC04- FramePoly
$1A04- ClosePort       $5004- Pt2Rect         $8604- Random          $BD04- PaintPoly
$1B04- SetPort         $5104- EqualRect       $8704- SetRandSeed     $BE04- ErasePoly
$1C04- GetPort         $5204- EmptyRect       $8804- GetPixel        $BF04- InvertPoly
$1D04- SetPortLoc      $5204- NotEmptyRect    $8904- ScalePt         $C004- FillPoly
$1E04- GetPortLoc      $5304- FrameRect       $8A04- MapPoint        $C104- OpenPoly
$1F04- SetPortRect     $5404- PaintRect       $8B04- MapRect         $C204- ClosePoly
$2004- GetPortRect     $5504- EraseRect       $8C04- MapRgn          $C304- KillPoly
$2104- SetPortSize     $5604- InvertRect      $8D04- SetStdProcs     $C404- OffsetPoly
$2204- MovePortTo      $5704- FillRect        $8E04- SetCursor       $C504- MapPoly
$2304- SetOrigin       $5804- FrameOval       $8F04- GetCursorAdr    $C604- SetClipHandle
$2404- SetClip         $5904- PaintOval       $9004- HideCursor      $C704- GetClipHandle
$2504- GetClip         $5A04- EraseOval       $9104- ShowCursor      $C804- SetVisHandle
$2604- ClipRect        $5B04- InvertOval      $9204- ObscureCursor   $C904- GetVisHandle
$2704- HidePen         $5C04- FillOval        $9404- SetFont         $CA04- InitCursor
$2804- ShowPen         $5D04- FrameRRect      $9504- GetFont         $CB04- SetBufDims
$2904- GetPen          $5E04- PaintRRect      $9604- GetFontInfo     $CC04- ForceBufDims
$2A04- SetPenState     $5F04- EraseRRect      $9704- GetFontGlobals  $CD04- SaveBufDims
$2B04- GetPenState     $6004- InvertRRect     $9804- SetFontFlags    $CE04- RestoreBufDims
$2C04- SetPenSize      $6104- FillRRect       $9904- GetFontFlags    $CF04- GetFGSize
$2D04- GetPenSize      $6204- FrameArc        $9A04- SetTextFace     $D004- SetFontID
$2E04- SetPenMode      $6304- PaintArc        $9B04- GetTextFace     $D104- GetFontID
$2F04- GetPenMode      $6404- EraseArc        $9C04- SetTextMode     $D204- SetTextSize
$3004- SetPenPat       $6504- InvertArc       $9D04- GetTextMode     $D304- GetTextSize
$3104- GetPenPat       $6604- FillArc         $9E04- SetSpaceExtra   $D404- SetCharExtra
$3204- SetPenMask      $6704- NewRgn          $9F04- GetSpaceExtra   $D504- GetCharExtra
$3304- GetPenMask      $6804- DisposeRgn      $A004- SetForeColor    $D604- PPToPort
$3404- SetBackPat      $6904- CopyRgn         $A104- GetForeColor    $D704- InflateTextBuffer
$3504- GetBackPat      $6A04- SetEmptyRgn     $A204- SetBackColor    $D804- GetROMFont
$3604- PenNormal       $6B04- SetRectRgn      $A304- GetBackColor    $D904- GetFontLore
$3704- SetSolidPenPat  $6C04- RectRgn         $A404- DrawChar
```

Fig. 2 – Numeric order list of IIgs tools (part 1) . . .

slow you down and might severely limit your choice of your host computer, besides costing you lots of extra money.

A reminder that yours truly and a bunch of other Apple incindiaries – uh, better make that luminaries – will be talking and seminaring and showing and telling at the stupendous AzApple Fiesta at the Safari Hotel in Scottsdale Arizona, on June 11th and 12th. For more details, you might contact Jerry Cline over at AzApple via (602) 264-3800.

And it looks like its time to try and keep *Canis Lupus* away from the door again. So, the usual reminders that we have bound sets of my *Ask the Guru* reprints available, along with my *Introduction to PostScript* video and my new *PostScript Show and Tell*, which is available for most popular personal computers.

Also some brand new PostScript language packages one for printed circuit layout and another for two point perspective drawing. Write or call if you are interested.

Let's start with a summer rerun . . .

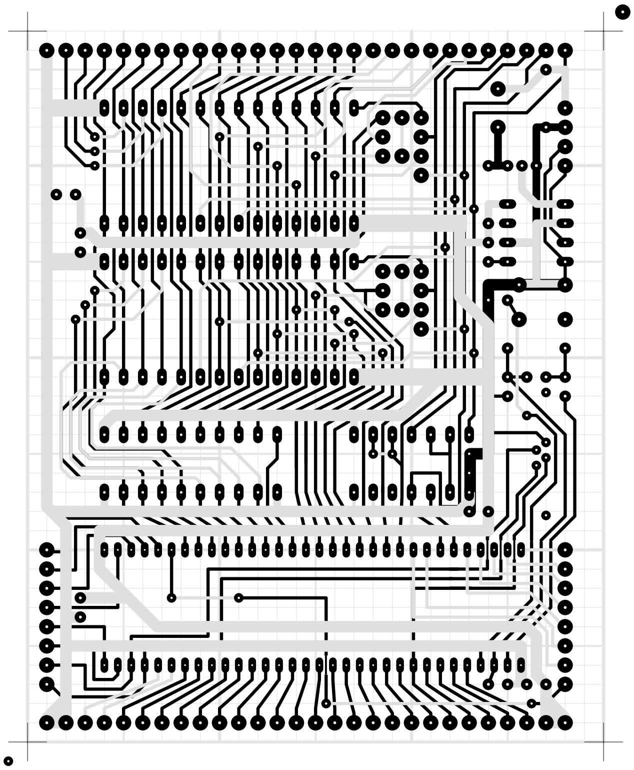


Fig. 3 – A typical PostScript 1:1 printed circuit layout.

How can I Use the Old 20 Pin Drives On a IIgs?

We have had a lot of calls on this recently, so let's do a rerun. Figure one shows you how to build a dual "old drive" adaptor that lets you daisy chain the original 20-pin 5-1/4 inch Apple drives onto the IIgs 3-1/2 drive 19 pin output connectors.

Note particularly that pin 4 must be grounded, and a pullup resistor is recommended as shown. Also be very careful to use shrouded or otherwise restrained connectors. If you wrongly plug a 5-1/4 drive in so it is offset by one pin or by one pin row, you can instantly destroy several drives and the IIgs motherboard.

Note also that, yes, you can freely plug printer cables in and out any time you like. But if you ever try connecting or disconnecting a disk drive on a IIgs with the power on, you are virtually *certain* to destroy either or both the drive or the IIgs motherboard.

Do NOT ever add or remove any drive when IIgs power is applied!

Remember finally that there is a "rundown" time on your IIgs power supply. Always wait a minimum of 40 seconds after turning the power off before you even think about changing any card or cable.

One commercial source of these adaptors is *Redmond Cable*.

Show me a List of The Main IIgs Tools

One of the best ways to learn to use the tools in the IIgs toolkit is to tear apart the working code of others to see just how it is done.

A toolbox call is made by shoving things onto the stack, loading the X register with a two hex digit high value for the tool and a two hex digit low value for the toolset. A JSL $E10000 is then done to activate the tool. The tool, in turn does its thing, makes some changes, and returns some pointers and results back to the stack for future use.

I have never found a really good "backwards" list of all the IIgs tools arranged in numeric order by call, so figure two gives you the start of a more or less complete list.

More details on each individual tool do appear in the *IIgs Toolbox Reference Manauls*, volumes I and II, available through the *APDA* people. And for a super-secret, ultra fast, and extremely simple method of tearing apart most any machine language program, check into my *Enhancing your Apple IIe*, volume I.

Which Laser Printer Should I Buy ?

That is a very good question that comes up at least a dozen times a day on the helpline.

Here are what the helpline callers seem to be thinking at present. First and foremost, genuine Adobe PostScript is an absolute must, since there is between a 50:1 and 100:1 performance advantage over all of the outdated imitators and hanger-ons.

Secondly, you might want to avoid the *Riccoh* engine, since helpline callers do report serious problems imaging smaller text, cite the slow speed, note the lack of a manual feed, and complain about the long term print reliability.

It seems to me that this leaves you with five reasonably viable options

options towards genuinely useful laser printing today . . .

(1) The Laserwriter NT or the NTX, which should be the "center of the universe" of PostScript laser printing for the next several years.

(2) The older Laserwriter plus is now being discounted for as low as $1800, with the latest of third-party toner refills now being almost as black as the newer machines.

(3) The brand new "laserless" *NEC LC-890* PostScript printer, which you helpline callers can't seem to say enough good things about.

(4) Any of the PostScript speaking laser printers from *QMS*.

(5) The *Hewlett-Packard Laserjet II*. But beware that this turkey provides PostScript only in a rather expensive and very bizarre manner that *demands* use of an IBM clone.

But you definitely should not buy any of these, until such time as you have had some hands-on experience with them running your material your way through your local user group.

I personally own a pair of LaserWriter pluses, one with over 300,000 copies on it and now have a brand new NTX. I just can't believe I was able to squeak by so long with only two PostScript speaking printers in my home. On the other hand, the light in the pantry *still* is not fixed.

I guess its all in the priorities. Does anyone know an electrician that needs some typesetting done?

What is Padding Compound?

Well, it comes in a large jar the same size and shape as the library paste you used to eat with Cecilia Winterhalter back in Mrs. Lockhart's second grade art class. It does cost around $5.50 a jar, and is available from *Paper Plus* or most any other printer supply house.

It does look sort of like Elmer's glue-all. You paint this glop onto the edge of a stack of paper, and end up with an instant notepad, fully professional prescription pads, a "while you were out" form, or even several custom calanders.

While not at all a replacement for any "real" binding system, padding compound is a quick and fun way of at least temporarily holding paper stacks together. At any rate, all of my beginning desktop publishing students at EAC sure are getting off on the stuff. For some reason, the perscription pads seem to be popular.

An ordinary paintbrush works just fine, and you can clean up with water before the padding compound sets. You should carefully align and trim all the pages first, and should apply some clamping pressure while your padding compound sets.

The stuff takes half an hour to dry, and two thin coats are usually better than one thick one. One jar is enough for a zillion and a half pads.

While nothing at all is needed in the way of special tools for just a few hand-crafted pads, to handle the job professionally, you might want to build a padding press, which looks sort of like a Fred Flintstone style enlarger. The use of a clamping paper cutter and a jogger are also handy production tools.

Note that many "jiffy" printing centers will do thick paper cutting for 50 cents a cut or so.

For minimum labor, it is often best to pad the two opposite edges of an entire stack of full size sheets at once; you then chop these up into four or six pads as needed. By alternating stacks of paper and the backing cardboard, you can work several pads thick.

Tellyawhat. For the contest this month, just dream up a brand new but non-obscene use for padding compound. There'll be an *Incredible Secret Money Machine* for the best twenty entries, and an all expense paid (FOB Thatcher, AZ) *tinaja quest* for two for the very best entry.

```
%  Copyright c 1988 by Don Lancaster & Synergetics, Box 809, Thatcher AZ, 85552.
%  (602) 428-4073. All rights reserved. Personal, non-commercial use permitted so long as
%  this header remains both present and intact. Work in Progress disk costs $39.50.

/quadpixel {transform 4 div round 4 mul itransform} def

/setgrid {save  /rubbersnap exch def /size exch def quadpixel exch quadpixel exch translate
size dup scale } def

/drawlines {72 300 div lw mul size div setlinewidth /hposs 0 def #hlines gs div 1 add cvi
{hposs 0 moveto 0 #vlines rlineto stroke /hposs hposs gs add def} repeat /vposs 0 def
#vlines gs div 1 add cvi {0 vposs moveto #hlines 0 rlineto stroke /vposs vposs gs add
def} repeat} def

/showgrid{gsave /#vlines exch def /#hlines exch def 106 45 {pop pop 0} setscreen 0.9
setgray /gs 1 def /lw 1 def drawlines  grestore} def

/2X {60 72 mul 300 div setgrid} def

/trace20 {6 30 div setlinewidth} def /trace50 {16 30 div setlinewidth} def /trace80 {24
30 div setlinewidth} def

/am {newpath moveto} def
/tdraw {rlineto currentpoint stroke moveto} def

/u {0 exch tdraw} def /r+ {dup tdraw} def
/r {0 tdraw} def  /r- {dup neg tdraw} def
/d {0 exch neg tdraw} def /l- {neg dup tdraw} def
/l {neg 0 tdraw} def /l+ {dup neg exch tdraw} def

/black {0 setgray} def /white {1 setgray} def

/xrpt{gsave aload pop /trips exch def /dist exch def /rproc exch def trips { gsave rproc
grestore dist 0 translate } repeat grestore} def

/yrpt{gsave aload pop /trips exch def /dist exch def /rproc exch def trips { gsave
rproc grestore 0 dist translate } repeat grestore} def

/hole {gsave 150 div /dia exch def newpath dia 2 div 0 360 arc white fill grestore} def

/icpad1v {save /psnap exch def trace50 2 copy gsave exch 0.2 sub exch am 0.4 r grestore
20 hole psnap restore} def

/edgeconu {gsave translate 0.4 0 moveto 0 -2 0.4 0 180 arcn 0 2 rlineto closepath fill
grestore} def

/feedpad {save /fpsnap exch def newpath 2 copy black 0.25 0 360 arc fill 18 hole clear
fpsnap restore } def

/circpad2 {save /fpsnap exch def newpath 2 copy black 0.30 0 360 arc fill 22 hole clear
fpsnap restore } def

/dip8v{gsave translate [{0 0 icpad1v} 1 4] yrpt [{3 0 icpad1v} 1 4] yrpt grestore} def

/dip16v{gsave translate [{0 0 icpad1v} 1 8] yrpt [{3 0 icpad1v} 1 8] yrpt grestore} def

1 setlinejoin 1 setlinecap
```

Fig. 4 – A sampler of my PostScript printed circuit routines.

What is This Month's PostScript utility?

For years, I have been searching for a practical and economical way for hackers and engineers to do low-end printed circuit layouts.

Obvious goals here are very high quality, low cost, use of nearly any computer at all, avoiding any and all outrageously expensive supplies, and an absolute minimum need for anything at all photographic.

I've been putting together a set of PostScript language printed circuit utilities that does this and a lot more. You can do 1X, 2X, and 4X layouts, frontwards or backwards, negative or positive artwork, with or without an overlay grid in ten or more layers in any size from one square inch up to several acres. Very nicely, you will always work "top side" and let your printer handle any trace reversals or inversions for you.

As per usual with "raw" PostScript, you can use any old word processor on any old personal computer to do the job. *AppleWriter*, of course, is an especially good choice. No special graphics tools or techniques are required, and the ordinary text files can then be freely passed over to any other computer.

The utilities are particularly good for such "mixed media" job tasks as an instruction book or a manual that needs "phantom" layouts for servicing or drilling guides. Best of all, the routines return "hand crafting" back to the world of printed circuit layout art. This is especially handy for high frequency analog or video circuits.

It should even be possible to now transfer toner directly to the copper pc board material for an immediate etching, and thus eliminating some intermediate photo steps. Sadly, the techniques I've looked at so far "just barely miss" on this additional goal, but do stay tuned.

And, of course, I would certainly welcome any and all input on a suitable transfer material that was stable, cheap, and reliable. There is a humongous market just waiting here for the right product. Once again, the material must be transparent, dimensionally stable at high temperatures, and has to "loosely" but reliably accept toner. The toner must then be able to be reliably transferred directly to a printed circuit board for use as an etch resist. Help!

```
% requires printed circuit sampler code of figure four

100 100 2X 11 11 showgrid trace20 10 2 am 1 l 0.75 l- 4.5 l 0.75 l+
2 l 6 u 10 5 am 0.75 l 1.5 l- 1.25 l 0.5 l- 2 l 0.5 l+ 0.75 u 1.25 l+ 7 4 am
0.75 l  0.5 l- 0.75 l 5 5.5 am 1 l+ 1.5 u 5 4.5 am 2 l+ 2 u 2 6.5 am 1.5
u 7 3 am 0.5 l- 2.75 l 0.75 l+ 0.5 u 0.75 l+ 2 3.5 am 1 l 10 9 am 0.75 l+
2.75 l 0.5 l- 2.75 d 1 l 6 6.5 am 2.15 d 0.35 l- 1.15 l 2 l+ 1 l 5 6.5 am
1.5 u 7 2 dip16v 2 3.5 dip8v [{1 10 edgeconu} 1 5] xrpt 3 2 circpad2
4 2 circpad2 3 5.5 feedpad showpage
```

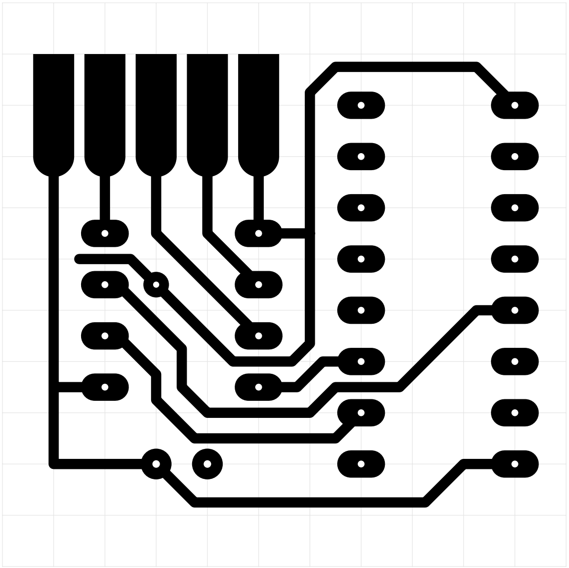


Fig. 5 – A simple 2X printed circuit layout example.

Figure three shows you a typical working layout. Here, viewed from the top of the board are an overlay grid, the component side traces in gray, and finally the foil side done in black. As you can see, arbitrarily complex layouts are simple and easy to do. This one is a $50 control computer using the M50734.

What I would like to do here is share several of the more fundamental PostScript language pc layout codes with you. Figure four is a "sampler" listing that can get you started, while figure five shows you a portion of the layout of a simple and sensitive 10-bit A/D converter that plugs directly into your Apple IIe or IIgs game paddle port. More details on this gem appear in my *Hardware Hacker* column that appears over in *Radio-Electronics* magazine.

The full blown utilities consist of three parts. A *prolog* contains the hundreds of available custom PostScript layout commands. A *print instructions* header then selects the scale, the reversals, the page position for larger printouts, and so on. And finally, your own *script* contains a listing of the pads and traces.

There is a limit to the total path length used in any PostScript routine. So, rather than defining a single seperate path for the grid, component side, foil side, and overlay callouts, these are allowed to be arbitrarily long and complex. In the complete utilities package, the print instructions header then picks either the page or the *nulldevice* for each layer. Anything printed to the nulldevice ends up intentionally missing on the chosen final artwork copies.

Ordinary polyester (mylar) overhead transparency material seems to work just fine for most hacker and low-end circuit layouts.

Let me know if you have any further interest in pc layout stuff. It sure is fun to play with, and it does outperform several very costly software packages.

Don Lancaster's

ASK THE GURU

July, 1988

The LAN of the eighties
IIgs numeric tool listing
Restoring ProDOS disks
Corner rounding punches
Laser "personal map" cards

If all computer programs were drugs, then *Hypercard* would undoubtedly be the software equivalent of crack. Cheaply available on any street corner, it is insanely addictive. It gives the junkie user a brief and extremely intense euphoria, followed by the usual hollow promises, and finally ending up with withdrawl symptoms that become crushingly devastating.

Hypercard also forces continual dosage increases on the user, all the while shortening the tolerable time between fixes. And finally, near the very end, extremely grave delusions of adequacy set in.

At any rate, like it or not, Hypercard and its imitators are coming soon to a IIc, IIe, or a IIgs near you. Since forewarned is forearmed, you may want to have a long talk with your children, as well as starting up your neighborhood action group to try and somehow cope with this societal menance.

Remember, just one single dose, and you are hooked for life. With no hope of a cure. And, no, Hypercard does not only strike those "bad" addict programmers across the tracks. Hypercard relentlessly strikes anywhere and everywhere without any warning at all, cutting across all walks of life.

Meanwhile, Apple has reduced the price of the IIgs. Well, sort of, anyway. What they really did was bundle up the expansion RAM card in with the IIgs, since the new operating system pretty much demands 512K just for itself. The part number for the new IIgs system is #A2P6024.

Oh, yeah. Some epsilon minus left the *strobe* line off the IIgs game paddle port. Contrary to the manuals, it is just plain not there. Which will severely limit some of the neat hardware expansion tricks that were easy to pull on the II+ and IIe.

On the other hand, if nobody noticed it for two years . . .

The good folks at *Adobe Systems* are up to all sorts of neat stuff. First, expect a total reworking of the PostScript ROM chips "real soon now". Which should give you yet another dramatic speed up. Their experience with their new display PostScript did show them several profound new tricks they could pull to reduce both the number and the intensity of the interpretive trips through the code.

Presumably, upgrade kits will soon be made available, similar to all of the previous ones. More details on this as they evolve.

PostScript has continuously been speeding up at roughly thirty percent per year. These days, most of that slowness still blamed on PostScript is really caused by such things as the ridiculous communications overhead (especially AppleTalk), the poor programming, stupid host protocols, and other dumb stuff like this.

Adobe has also formed a new third party developers group, modeled along the lines of all those certified developer programs of *Apple* and *Microsoft*. Also, as figure one shows us, they now have an extensive new techncal literature catalog. Some of their available free publications do appear in figure one.

Meanwhile, lots is happening on the toner cartridge reloading front. Several third-party people are now intensely attacking the high costs and the limited refillability of those *SX* cartridges, as demanded by the new LaserWriters.

Both the new NT and NTX LaserWriters still have an insanely higher per-page toner cost for most users than do the older LaserWriters.

The latest refilling secret is to find a beastie called a *#3 Unibit*, made by *Vise Grip*, and available from *Jensen Tools*, as well as most larger hardware stores. This dude lets you drill, rather than melt the refilling holes in either the SX or CX cartridges.

Tips on Writing PostScript Language Drivers (POST01)
An Optimization Case Study (POST02)
How to Avoid Device Specific PostScript Programs (POST03)
Simple Text Setting Calculations for PostScript Output (POST04)
PostScript Language Supplement (POST05)

Screen Fonts Licensing (FONT01)
AFM Interchange Format for PostScript Font Metrics (FONT02)
Supporting Downloadable PostScript Fonts (FONT03)
ABF Files: Binary Format for PostScript Screen Fonts (FONT04)
Macintosh FONT Resources (FONT 05)

Document Structuring Conventions 2.0 (FORM01)
Encapsulated PostScript File Format (FORM02)
PostScript Printer Description File Specification (FORM03)
Adobe Illustrator Document Specifications (FORM04)

PostScript LaserWriter Plus Update (SUPP01)
PostScript Linotron 100 v42 Update (SUPP02)
PostScript QMS PS800 Update (SUPP03)
PostScript DEC, VAX, SUN, and UNIX Update (SUPP04)
PostScript Agfa P400PS Update (SUPP05)
PostScript TI OmniLaser 2115 Update (SUPP06)
PostScript NEC LC-890 Update (SUPP07)

Note on Flow Control and Serial Communication (COMM01)
LaserWriter Serial I/O Patch (COMM02)
Developer Tools Diskette (DEV01)
LaserWriter Plus AFM Files (DATA 01)
PPD Printer Description Files (PPDF01)

Fig. 1 – Adobe Systems tech literature on PostScript.

You'll want to work upside down to keep chips out of the toner filling tank, but that Unibit will instantly give you a clean hole. Use a slow drilling speed.

And you can now plug up the hole with a plain old nickel *Caplug* that is available from the *Caplug* people, or their *Alliance Plastics* competitors.

You might want to press your fast forward button, because it is now – bleep – advertorial time! Complete sets of the *Ask the Guru* reprints do remain available, as do my various PostScript products, including that *Introduction to Postscript Video*, the *PostScript Show and Tell*, and the pre-release on my brand new *Don Lancaster's PostScript Secrets* book and disk combo. Yes, we also stock the new Adobe green book on PostScript program design.

Let us start off with an obscure history lesson . . .

Tell me all About The LAN of The Eighties

There sure is a lot of interest these days in LAN's, or *local area networks*. Many of these are poorly performing, insanely expensive, or grossly limiting in one way or another. But one LAN is clearly head and shoulders above all of the others. So much so, that I like to call it the *LAN of the eighties*.

The LAN of the eighties is a simple token ring loop. It requires only one single wire between stations. That wire need not be shielded or twisted pair, and even *bare* wire has been used in several of the tens of thousands of world-wide installations.

The LAN loop can be ten or more miles long. While only a few dozen servers is the norm, many hundreds can be installed. In turn, each server node normally is able to handle as many as several hundred users.

With the LAN of the eighties, each node uses a unique i.d. in a collision avoidance, token passing protocol. The node first checks to see if other traffic is present. If not, its very own uniquely coded packet will get transmitted. Each packet is then repeated several times to get the best possible error detection and correction.

Each LAN node is also sophisticated enough that it continuously measures the network's signal to noise ratio. Should any communications problems develop, then an alternate signalling route is automatically selected.

The ProDOS catalog entries begin on block $02 and continue from there. The simplest way to locate successive catalog entries is by their ASCII filename character strings.

BYTE +$00 - holds the storage type in its upper four bits and the length of the filename as the lower four bits. Common storage types include $00 for deleted; $01 for a single data block file; $02 for a 2-256 data block file; and $0D for a subdirectory.

BYTES +$01-0F - holds the filename in low ASCII, up to 15 characters.

BYTE +$10 - holds the file type. The file types include $04 for text, $06 for binary; $0F for directory; $19 for Appleworks data base; $1B for Appleworks word processing; $1B for Appleworks spreadsheet; $FC for Applesloth Basic, and $FF for a ProDOS system file.

BYTES +$11-12 - hold the block number of the key block for the file. This is arranged low byte first.

BYTES +$13-14 - hold the total number of blocks used. This is also arranged low byte first.

BYTES +$15-17 - hold the total length of the file, arranged in low byte - mid byte - high byte order.

BYTE +$1E - holds the access bits for the file. Bit $01 allows reading of the file. Bit $02 allows writing to the file. Bit $40 alows renaming, and Bit $80 permits destruction.
(lock = 0; unlock = 1)

Fig. 2 – Important Bytes in a ProDOS catalog entry.

Each node is extremely rugged and requires very little maintenence. The nodes use zero electrical input power, instead substituting an ingeneous kinetic energy transfer mechanism.

Supporting the LAN of the eighties is a streaming tape drive using very low cost media that continuously and permanantly will record any and all traffic. Thus, any message can be replayed at any time.

On the LAN of the eighties, the operator training is extremely fast and ridiculously simple. Even a user in a very high stress environment can master the entire LAN workings in approximately five seconds, since the entire system is hardware based. Yes, even the boss can learn to use it.

The LAN of the eighties has been thoroughly tested and debugged. So much so, that the total number of user hours to date exceeds that of *all* other networking schemes combined.

Yes, this network is so good that it is clearly the LAN of the eighties.

The *eighteen* eighties!

I am, of course, talking about the *Gamewell* fire alarm telegraph, otherwise known as that mangy red box scunging away on the pole down the street. Patented over a century ago.

The call boxes form an amazing electromechanical network that both addressed and solved today's LAN networking problems a century ahead of their time. Deja Vu, anyone?

How can I Reconstruct A Blown ProDos Disk?

The best way of all, of course, is to never let the disk blow up in the first place. Didn't anyone ever tell you about backup copies? Or *never* using an original disk for anything ever? Or keeping all your drives clean and at the correct speed?

Or that car dashboards get warm in the summertime? Or *never* putting a disk *anywhere* except into a drive or *immediately* back into its own protective envelope? Or *never* allowing smokers in the same bulding as your disks, let alone in the same room?

Anyway, sooner or later you may have to try and revive a sick or dead

```
----------------------------------- TOOL #05 - DESK MANAGER ----------------------------------
$0105- DeskBootInit      $0A05- RestScrn         $1405- GetDAStrPtr      $1B05- GetDAStrPtr
$0205- DeskStartUp       $0B05- SaveAll          $1505- OpenNDA          $1C05- CloseNDAByWinPtr
$0305- DeskShutDown      $0C05- RestAll          $1605- CloseNDA         $1D05- CloseAllNDAs
$0405- DeskVersion       $0E05- InstallNDA       $1705- SystemClick      $1E05- FixAppleMenu
$0505- DeskReset         $0F05- InstallCDA       $1805- System Edit
$0605- DeskStatus        $1105- ChooseCDA        $1905- System Task
$0905- SaveScrn          $1305- SetDAStrPtr      $1A05- SystemEvent
---------------------------------- TOOL #06 - EVENT MANAGER ----------------------------------
$0106- EMBootInit        $0906- DoWindows        $0F06- WaitMouseUp      $1506- FlushEvents
$0206- EMStartUp         $0A06- GetNextEvent     $1006- TickCount        $1606- GetOSEvent
$0306- EMShutDown        $0B06- EventAvail       $1106- GetOblTime       $1706- OSEventAvail
$0406- EMVersion         $0C06- GetMouse         $1206- GetCaretTime     $1806- SetEventMask
$0506- EMReset           $0D06- Button           $1306- SetSwith         $1906- FakeMouse
$0606- EMStatus          $0E06- StillDown        $1406- PostEvent
------------------------------------ TOOL #07 - SCHEDULER ------------------------------------
$0107- SchBootInit       $0307- SchShutDown      $0507- SchReset         $0907- SchAddTask
$0207- SchStartUp        $0407- SchVersion       $0607- SchStatus        $0A07- SchFlush
---------------------------------- TOOL #08 - SOUND MANAGER ----------------------------------
$0108- SoundBootInit     $0608- SoundToolStatus  $0D08- SetSoundVolume   $1208- SetSoundMIRQV
$0208- SoundStartUp      $0908- WriteRamBlock    $0E08- FFStartSound     $1308- SetUserSoundIRQV
$0308- SoundShutDown     $0A08- ReadRamBlock     $0F08- FFStopSound      $1408- FFSoundDoneStatus
$0408- SoundVersion      $0B08- GetTableAddress  $1008- FFSoundStatus
$0508- SoundReset        $0C08- GetSoundVolume   $1108- FFGeneratorStatus
----------------------------------- TOOL #09 - DESKTOP BUS -----------------------------------
$0109- ADBBootInit       $0609- ADBStatus        $0E09- SyncADNReceive   $1309- SetAbsScale
$0209- ADBStartUp        $0909- SendInfo         $0F09- AbsOn            $1409- SRQPoll
$0309- ADBShutDown       $0A09- ReadKeyMicroData $1009- AdsOff           $1509- SRQRemove
$0409- ADBVersion        $0B09- ReadKeyMicroMem  $1109- ReadAbs          $1609- ClearSRQTable
$0509- ADBReset          $0D09- AsyncaADBReceive $1209- GetAbsScale
----------------------------------- TOOL #0A - SANE TOOLSET ----------------------------------
$010A- SANEBootInit      $040A- SANEVersion      $060A- SANEStatus       $0A0A- SANEDecStr816
$020A- SANEStartup       $050A- SANEReset        $090A- SANEFP816        $0B0A- SANEElems816
$030A- SANEShutDown
------------------------------- TOOL #0B - INTEGER MATH TOOLS --------------------------------
$010B- IMBootInit        $0E0B- FixRatio         $190B- LoWord           $240B- Hex2Int
$020B- IMStartUp         $0F0B- FixMul           $1A0B- Long2Fix         $250B- Hex2Long
$030B- IMShutDown        $100B- FracMul          $1B0B- Fix2Long         $260B- Int2Dec
$040B- IMVersion         $110B- FixDiv           $1C0B- Fix2Frac         $270B- Long2Dec
$050B- IMReset           $120B- FracDiv          $1D0B- Frac2Fix         $280B- Dec2Int
$060B- IMStatus          $130B- FixRound         $1E0B- Fix2X            $290B- Dec2Long
$090B- Multilply         $140B- FracSqrt         $1F0B- Frac2X           $2A0B- HexIt
$0A0B- SDivide           $150B- FracCos          $200B- X2Fix
$0B0B- UDivide           $160B- FracSin          $210B- X2Frac
$0C0B- LongMul           $170B- FixATan2         $220B- Int2Hex
$0D0B- LongDivide        $180B- HiWord           $230B- Long2Hex
----------------------------------- TOOL #0C - TEXT TOOLS ------------------------------------
$010C- TextBootInit      $0C0C- GetInGlobals     $150C- InitTextDev      $1E0C- TextWriteBlock
$020C- TextStartUp       $0D0C- GetOutGlobals    $160C- CtlTextDev       $1F0C- ErrWriteBlock
$030C- TextShutDown      $0E0C- GetErrGlobals    $170C- StatusTextDev    $200C- WriteCString
$040C- TextVersion       $0F0C- SetInputDevice   $180C- WriteChar        $210C- ErrWriteCSrting
$050C- TextReset         $100C- SetOutputDevice  $190C- ErrWriteChar     $220C- ReadChar
$060C- TextStatus        $110C- SetErrorDevice   $1A0C- WriteLine        $230C- TextReadBlock
$090C- SetInGlobals      $120C- GetInputDevice   $1B0C- ErrWriteLine     $240C- ReadLine
$0A0C- SetOutGlobals     $130C- GetOutputDevice  $1C0C- WriteString
$0B0C- SetErrGlobals     $140C- GetErrorDevice   $1D0C- ErrWriteString
-------------------------------- TOOL #0E - WINDOW MANAGER -----------------------------------
$010E- WindBootInit      $180E- TrackGoAway      $2D0E- SetWFrame        $410E- SetDataSize
$020E- WindStartUp       $190E- MoveWindow       $2E0E- GetStruckRgn     $420E- GetMaxGrow
0030E- WindShutDown      $1A0E- DragWindow       $2F0E- GetContentRgn    $430E- SetMaxgrow
$040E- WindVersion       $1B0E- GrowWindow       $300E- GetUpdateRgn     $440E- GetScroll
$050E- WindReset         $1C0E- SizeWindow       $310E- GetDefProc       $450E- SetScroll
$060E- WindStatus        $1D0E- TaskMaster       $320E- SetDefProc       $460E- GetPage
$090E- NewWindow         $1F0E- EndUpdate        $330E- GetWControls     $470E- SetPage
$0A0E- CheckUpdate       $1E0E- BeginUpdate      $340E- SetOriginMask    $480E- GetContentDraw
$0B0E- CloseWindow       $200E- GetWMgrPort      $350E- GetInfoRefCon    $490E- SetContentDraw
$0C0E- Desktop           $210E- PinRect          $360E- SetInfoRefCon    $4A0E- GetInfoDraw
$0D0E- SetWTitle         $220E- HiLiteWindow     $370E- GetZoomRect      $4B0E- SetSysWindow
$0E0E- GetWTitle         $230E- ShowWide         $380E- SetZoomrect      $4C0E- GetSysWFlag
$0F0E- SetFrameColor     $240E- BringToFront     $390E- RefreshDesktop   $4D0E- StartDrawing
$100E- GetFrameColor     $250E- WindNewRes       $3A0E- InvalRect        $4E0E- SetWindowIcons
$110E- SelectWindow      $260E- Trackzoom        $3B0E- InvalRgn         $4F0E- GetRectInfo
$120E- HideWindow        $270E- ZoomWindow       $3C0E- ValidRect        $500E- StartInfoDrawing
$130E- ShowWindow        $280E- SetWRefCon       $3D0E- ValidRgn         $510E- EndInfoDrawing
$140E- SendBehind        $290E- GetWRefCon       $3E0E- GetContentOrigin $520E- GetFirstWindow
$150E- FrontWindow       $2A0E- GetNextWindow    $3F0E- SetContentOrigin $530E- WinddragRect
$160E- SetInfoDraw       $2B0E- GetWKind         $400E- GetDataSize      $560E- WindowGlobal
$170E- FindWindow        $2C0E- GetWFrame
```

Fig. 3 – Numeric order list of IIgs tools (part 2) . . .

ProDOS disk. The better you get at this, the bigger a hero you'll become.

Two important books include the *ProDOS Techncal Reference Manual* by *Addison Wesley*, and *Beneath Apple ProDOS* by *Quality Software*.

Reconstruction software includes the *Copy II Plus* from *Central Point Software*, and *Bag Of Tricks II*, again from *Quality Software*.

The first thing to try is a fresh reboot of your machine, followed by several repeated attempts to read the disk. Then try the other drive. If that doesn't hack it, try a different computer. A disk may get flakey before it gets bad. Obviously, you will want to transfer the data any way you can.

The very next thing to check for is obvious physical damage. If the disk got tricycled by a three year old or run over by an office chair, it can often be repaired by cutting the top off the disk and inserting it in a new disk carrier, preferably of the same brand and style. Don't touch the mylar media when you do this.

If a disk has been re-initialized, then all is lost. You can run over a disk with a truck and then boil both the disk and the truck in peanut butter and jelly, and the chances are you can recover some of the data.

But Init kills! So, do not *ever* init a disk without first removing it from the drive, holding it by the write protect tab, and spelling the name out *backwards* and out loud. With all other drive doors open. *Never* copy a disk without write protect tabs.

Remember that 5-1/4 inch disks protect *backwards* from the 3-1/2 inch ones. An *open* notch on a 5-1/4 disk can be written, while a *closed* window on a 3-1/2 is also unsafe.

You should now make the best bit copy backup you can of the original disk. Don't EVER attempt to reconstruct the actual bad disk. Instead, *always* work on a bit copied backup.

Then, try using a disk recovery utility, such as *Fixcat* from *Bag of Tricks*. Since catalog tracks are the ones most often written to, they tend to blow up the most often.

If all else fails, a partial manual recovery may be all that's left to try.

To get started, load up a *Zap* utility or some other program that lets you read and write valid ProDOS blocks. Then carefully go through the entire backup disk by hand, reading each and every block by hand, and taking lots of notes on everything you find.

First, find out the bad blocks that will not even read. List each of these. You will have to patch your way around these, and you can probably write off any data present, unless you go to some very exotic tools.

Next, be on the lookout for *key blocks*. A key block will have mostly zeros. Most often, it will start out with a sequence such as: 34 36 37 38 39 3A 00 00 00 . . . These are the blocks where the file will reside.

The high block number byte is 256 locations further along. The high byte will often be 00.

Once you have a key block, try to figure out what type of file it is and how important it is that you recover it. In general, you only want to try and recover what is essential. Some early files may later get overwritten. If the block data takes a "right angle turn" in the middle, or if later key blocks usurp the same data blocks, then the file is no longer used.

The object of all this should be a list of the files you want to recover. Should one block be blown up in the middle of a file, replace that block number in the key block with the previous block number. This way, you will get 512 good but repeat bytes replacing any bad block.

If the catalog fixer did not work, you'll have to manually patch a new catalog to link what you have with what you want. So, init a new disk, and create dummy files AAA through ZZZ on it. Then very carefully copy track zero and *only* track zero from this new disk to the backup.

Reread block two of the backup. Your new catalog should be there. Then, consulting figure two, or one of the ProDOS books, manually patch each catalog entry as needed.

In general, you will have to patch byte +$00 of a directory entry with the storage type, and byte +$10 with the file type to be recovered.

Bytes +$11 and +$12 get filled with the block number of the recovery key block. Values go in place *low byte first*, as is typical with 6502 code.

You will also have to patch +$13 and +$14 with the total number of blocks used in the recovery file.

For the next step, you'll have to calculate the exact file length of the file to be recovered and put it in +$15 (low) +$16 (mid) and +$17 (high).

Finally, attempt to read your AAA entry and see if it actually loads your recovered file. If it does, *immediately* transfer this recovered file to a clean disk. Then rename the file.

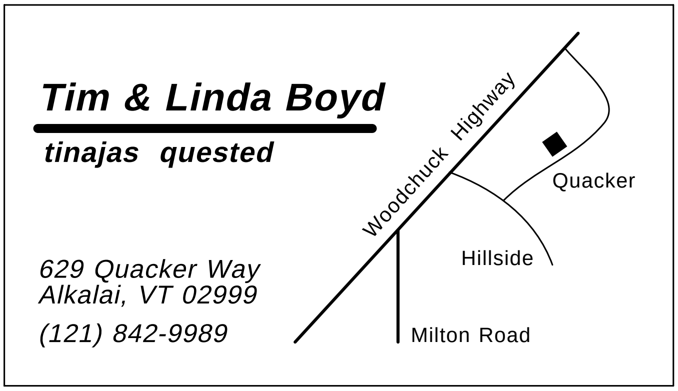
This process really sounds hairy, but if you pick up the needed books and tools and practice it bunches ahead of time, disk recovery can be made into a non-gruesome process.

Show me More of The Main IIgs Tool List

As we found out last month, a complete listing of the IIgs tools in numeric order is quite hard to find, yet is extremely useful for analyzing other people's IIgs code. Our second installment appears in figure three.

Please Help me Cut Some Corners

Sure thing. As we learned before, there is a never ending assortment of



Tim & Linda Boyd

tinajas quested

629 Quacker Way
Alkalai, VT 02999

(121) 842-9989

Fig. 4 – A typical "personal map" business card.

overpriced and hard-to-get, tools that can make desktop publishing and its laser printing ridiculously more powerful and more useful. Stuff like joggers, folders, laminators, drilling machines, padding presses, fusers, guillotine cutters binders, and so on.

At any rate, the chances are you might like to round off the corners of such things as bumperstickers, shipping labels, punched card coupons, VHS cassette labels, peel-and-stick stuff, point-of-purchase signs, menus, or just about whatever.

So, there is now a beastie called the *Lassco Corner Rounder*, that is manufactured by *Lassco Products*. This will round a stack of sheets at once, and accepts various dies, with 1/8 inch and 1/4 inch being the most popular. The list price is $120, but these sometimes get discounted down into the $85 to $95 range.

As with virtually all of the other accessory salesmen for the printer trade, you'll find most of the Lassco dealers grossly incompetent, ridiculously overpriced, and unbelievably rude. Lassco, of course, will tell you with a perfectly straight face to "see your dealer".

By the way, if you're only interested in printing Beta or VHS box, spline, and face video labels, several cheap and attractive blanks are now available from the *Polyline Corp*.

And if you need great heaping bunches of custom labels or label blanks of most any size, you might try my neighbor down the street, *Hy-Tech Identification Products*.

What is this month's PostScript utility?

Well, this month I thought I would bring you up to date on the pitfalls and opportunities of laser printed business cards and similiar items.

Firstoff, there is no way you can compete one-on-one with the existing *standard layout* card sources. Several loss-leader ads in the *Wall Street Journal* will give you one thousand raised-print cards for around $8. And there is also no way that toner can be made as durable as regular ink, let alone raised print thermography.

The first unique thing you might offer is immediate service on totally custom business cards, delivered here and now. The second thing you can offer is a very low total price, say $5 for 48 cards. The third thing you can offer is virtually any graphic image at all on their custom card.

For instance, how many printers can sell you custom business cards that have maps on them? Yet there are literally millions of people who'd like to buy a few dozen "Here's how to get to our house" cards for their own personal or family use. Most any swap meet or fair should be full of them. Figure four does show you a typical example.

And a fourth thing you can offer is unusual finishes and styles. You might use a fancy mother-of-pearl card stock, *Kroy Kolor* it a bright metallic blue, and seal it in plastic, for around a dime a card. *Coburn* is one source of unusual materials.

The durability of the toner can be improved in several ways. First, it is probably good enough by itself for those quick and dirty "throwaway" cards that are distributed in larger quantities. But a toner business card will self-destruct if it is carried in a wallet or is otherwise scuffed.

The simpliest way to improve the durability of toner is to run it back through the printer a second time, iron it, or send it through a Kroy Kolor machine. If you do this while in contact with a thin sheet of mylar (such as an "empty" Kroy Kolor carier), the image will *Bakerize*, and both darken and pick up a high gloss.

You can also spray your cards with artists fixative, but Bakerizing does a better job and costs less. Kroy Color itself will slightly increase the durability. The ultimate trip, though, is to use the Kroy Laminating Film to create a virtually unscuffable card.

Normally, you would use plain old cover stock in various colors as the business card base. Typically, you can get a dozen cards per sheet. The *Paper Plus* chain is a low cost source for cover stock.

The PostScript code for a typical step-and-repeat personal map card is shown you in figure five. You'll also need my step-and-repeat routine, my curve tracing routine, and rubbergrid.

You can get these from back issues of *Computer Shopper*, from the *Ask The Guru* reprints, or from my new *Don Lancaster's PostScript Secrets* book and disk combo, or else I'll be happy to send you a free printed copy of each needed routine.

We'll have both a technical and a non-technical contest for this month. There will be the usual *Incredible Secret Money Machine* prizes, with an all expense paid (FOB Thatcher, AZ) *tinaja quest* for two for the best. You can either send me an unusual card design that uses my PostScript code, or else show me a practical way to combine both the toner and the thermographic ("raised ink") technologies together into one single product. Let's hear from you.

```
% "personal map" 12-up business card PostScript example
%  . . . . . . . .

%  requires Don Lancaster's step-and-repeat, rubbergrid, and curvetracing
%  modules, listed elsewhere in these reprints

/font1 {/Helvetica-BoldOblique findfont [1.5 0 0 1.5 0 0] makefont setfont} def
/font2 {/Helvetica-BoldOblique findfont [1.1 0 0 1.1 0 0] makefont setfont} def
/font3 {/Helvetica-Oblique findfont [1 0 0 1 0 0] makefont setfont} def
/font4 {/Helvetica findfont [0.8 0 0 0.8 0 0] makefont setfont} def

/bakercard {0 0 10 setgrid  25 14 showgrid font1 2 10 (Tim  &  Linda  Baker)
cl font2 1.8  8 ( tinajas  quested ) cl gsave 2 9.3 m 0.35 setlinewidth 1
setlinecap 13 r gsave font3 2 3.5 (629 Quacker Way) cl 2  2.5
(Alkalai, VT 02999 ) cl 2 1 ((121) 842-9989) cl line2 12 1 m 23 13 lineto
stroke 16 1 m 4.35 u gsave 21.5 8.2 m 40 rotate 0.7 0 rlineto 0 0.7 rlineto
-0.7 0 rlineto closepath fill grestore line1 [18 7.6 -20  22 4 -70] curvetrace
stroke [20.1 6.5 45  24 9.5 50  22.5 12.4 130] curvetrace stroke font4 16.5 1
(Milton Road) cl 21.3 4 (Hillside) cr 22 7 (Quacker) cl gsave 15 5 translate
48 rotate 0 0 (Woodchuck  Highway) cl grestore grestore} def

/repeatproc {bakercard} def  (businesscard)  stepandrepeat
```

Fig. 5 – PostScript code for the "personal map" cards.

Don Lancaster's

ASK THE GURU

August, 1988

IIgs numeric toolkit listing
An exact IIgs video locking
Post-justification edit ideas
Handicapped tech resources
Postscript input text scanner

Well, as that zookeeper said, we have got lots of gnus this month. Apple is finally at the kicking and screaming stage of putting the final touches on their new *Personal AppleLink* network, which should now provide direct end-user support at the corporate level.

This will add to and supplement the already existing *AppleLink* network that is now available to the user groups, developers, and the dealers. Apple end user support, of course, is long overdue.

Meanwhile, in a totally unrelated development, IBM has already introduced a new stunningly spectacular end-user corporate level resource that's named *IBM Link*. They are to be congratulated on all of their progressive thinking and their ground breaking innovation here.

Especially in their uniquely original name choice.

The *Weitek* people have now just completely updumped the *PostScript* applecart by coming out with a new hardware based RISC microcomputer that directly is able to execute all its PostScript commands ten to thirty times faster than the *LaserWriter NT*. And do so at lower cost. That's ten to thirty *times*, not ten to thirty percent.

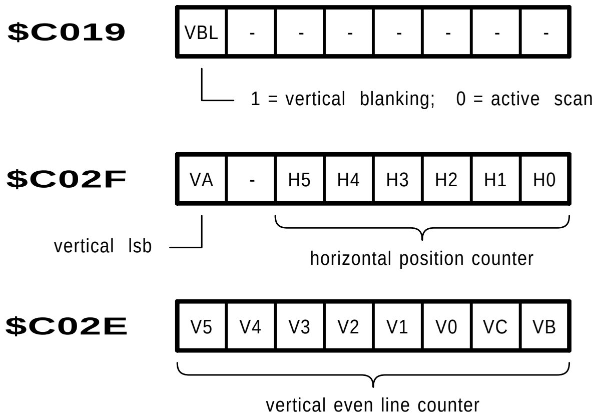
The speedup mainly lies in the difference between self-compiling hardware versus interpreting software. A dedicated hardware engine is almost always much faster than using general purpose interpreted software. But, obviously, not nearly as flexible. *Weitek* apparently chose a middle ground here, in having such goodies as BitBlt transfers and cubic spline generation done rapidly in hardware, while they still software interpret other PostScript commands.

So far, the beast is neither fish nor fowl, since it does not have the endorsement of either *Adobe* or *Apple*. Those "also-rans" are sure glomping onto this jewel, though.

Apple continues to keep their head in the sand on this, laboring under the delusion that *QuickDraw* is in some manner remotely useable, while completely ignoring all of the overwhelming advantages of standard display PostScript. Nobody bothered to tell them that QuickDraw never had been, is not now, and is highly unlikely to ever become, a viable page description language.

The laser printers that use QuickDraw instead of PostScript are all an outright joke. Not to mention, in my humble opinion, being a total ripoff.

Hopefully, though, at least one third party should soon have display PostScript available for the Mac and the IIgs, possibly including a hardware speedup engine. More on this whenever and wherever.



$C019
VBL
1 = vertical blanking; 0 = active scan
$C02F
VA
H5
H4
H3
H2
H1
H0
vertical lsb
horizontal position counter
$C02E
V5
V4
V3
V2
V1
V0
VC
VB
vertical even line counter

Fig. 1 – Those secret IIgs screen locking locations.

Details on writing your own IIgs port and print drivers now appear in the *Apple IIgs Technical Notes* #36 and 37. These should be available through the usual BBS and your user group library sources. As always, you can now get a listing of your local Apple and other user groups right here in *Computer Shopper*, or else call (800) 538-9696, Extension 500.

There is also a special new users supergroup for club ambassadors. It is called *AAIN*, and is short for *Apple Ambassador's Information Network*.

Meanwhile, there's an outfit called *K-12 MicroMedia* that has bunches of *AppleWorks* programs and accessories available. You might like to give them a call for more info.

There is absolutely no truth to the rumor that the super-secret new IIgs upgrade is so fast that it can execute an infinite loop in one minute and seventeen seconds flat.

And the winnar is . . . Dale Rice of Lansing, Michigan does win the all expense paid (FOB, Thatcher AZ) tinaja quest for two in our padding compound contest.

To quote Dale "First, go to the bank and request 40 or 50 brand new uncirculated one dollar bills. It is super important to get them in sequence to insure they are crisp and uncirculated. Next, use the padding compound and some cardboard to create a note pad of the bills. Be sure to include a "now is the time to reorder" slip near the stack bottom.

Now, take your pad along to your favorite bar or restaurant, and when it is time to pay, peel off the bills from your George Washington notepad. It is better than candid camera!"

Other entrants did suggest using padding compound to bulletproof Easter eggs and other fragile craft items, and pointed out that ordinary food dye can be used to get other colors besides the stock red or white.

Did I ever tell you the one about my *Ask the Guru* reprints, or my *Postscript Show and Tell*, or my new *Don Lancaster's PostScript Secrets* book and disk combo for all major personal computers? Or all about my *Hardware Hacker* column that you might find over in *Radio-Electronics* magazine?

As per usual, this is your column and you can get tech help and off-the-wall networking per the end box.

Our feature distraction this month is an exitable PostScript text scanner. But first, let's find out . . .

How can I do a IIgs Screen Lock?

On the original Apple II, there was no immediate or obvious way to tell where the video circuitry happened to be scanning at any particular instant. This produced badly torn animation, screen glitches, and other unpleasantries that usually gave you a less than optimum video display.

Eventually, some sneaky hardware and software tricks were discovered by Bob Bishop, by myself, and by others that let you do an effective screen lock with your software. This, in turn led you to smoother screen changes, to glitchless animation, and the ability to mix text, HIRES, and LORES graphics in any combination anywhere on the screen, real-time windowing, improved color killing, gentle scrolling, video wipes and fades, and, in general, all-around more professional video results.

Much of this is now ancient history. Full details do appear in my *Enhancing your Apple II and IIe*, volumes I and II.

The IIgs has three new resources that make doing screen locks of most any complexity much easier and far more powerful. Figure one bares all.

Machine language location $C019 has a simple vertical blanking flag in its most significant bit location. If this bit is a "1", then the vertical retrace is taking place and anything you do to the screen will currently remain invisible.

If you do wait until the vertical blanking interval before changing any video mode, you might get a glitchless change without any screen tearing or flashing. Note that the sense of $C019 is the *opposite* of that on the IIe. Watch this detail.

Machine language location $C02E will let you directly read the vertical line counting hardware. By changing the top of the screen while scanning the bottom, and vice versa, you can now do glitchless animation. By the way, the cause of many glitches is giving the viewer a simultaneous mix of the "old" and "new" information during a field.

Note that there are *nine* vertical counter bits. The least significant of these appears in $C02F, along with the horizontal counter bits. Thus, $C04E by itself will only read *even* vertical lines.

Finally, that secret machine language location at $C02F lets you directly read the horizontal line counter. Since the horizontal counts do change at a one microsecond clip, you have to be super tricky to be able to use these horizontal counts for anything at all useful.

On the other hand, an exact screen video lock can be done by repeatedly reading the horizontal line count with sneaky enough software. Which is

Apple Special Education
20525 Mariani Avenue 36-M
Cupertino, CA 95014
(408) 996-1010

Children's Resource Center
1056 E 19th Avenue
Denver, CO 80218
(303) 861-6633

Closing the Gap, Inc.
PO Box 68
Henderson, MN 56044
(612) 248-3294

Communication Resources
3201 Marshall Road
Kettering, OH 45429
(513) 298-0803

Computer Access Center
2425 16th Street Room 23
Santa Monica, CA 90405
(213) 450-8827

Computer CITE
215 East New Hampshire
Orlando, FL 32084
(305) 299-5000

Children's Computer Group
2095 Rose Street
Berkeley, CA 94709
(415) 841-3224

Disabled Children's Group
1146 South Third Street
Louisville, KY 40203
(502) 584-1239

Disabled Technical Center
5759 Hedgehaven Court
Las Vegas, NV 89120
(702) 382-3358

Handi-Ham System
3915 Golden Valley Road
Golden Valley, MN 55422
(612) 588-0811

National Braille Press
88 Saint Stephen Street
Boston, MA 02115
(617) 266-6160

Pacer Center
4826 Chicago Ave South
Minneapolis, MN 55417
(612) 827-2966

SpecialLink
2512 Canterbury Avenue
Cincinnati, OH 45212
(513) 531-9233

Special Technical Center
Route 4, 433 East Lafayette
Jackson, TN 38301
(901) 423-9058

Special Technology Center
535 Race St, Suite 220
San Jose, CA 95126
(408) 288-5010

SuperGroup Evaluation
4129 Beaujolais
Kenner, LA 70065
(504) 561-8713

Technical Access Center
183 Lake Avenue
Newton, MA 02159
(617) 969-4279

Technical Assistance
1950 West Roosevelt Road
Chicago, IL 60608
(312) 421-3373

Technology Resources
3023 Canterbury
Salina, KS 67401
(913) 827-0301

Trace Development Center
1500 Highland Avenue
Madison, WI 53705
(608) 262-6966

Fig. 2 – Some technical resources for the handicapped.

TOOL #0F - MENU MANAGER

$010F-	MenuBootInit	$110F-	GetSysBar	$1F0F-	SetMenuFlag	$2D0F-	NewMenu
$020F-	MenuStartUp	$120F-	SetSysBar	$210F-	SetMenuTitle	$2E0F-	DisposeMenu
$030F-	MenuShutDown	$130F-	FixMenuMItem	$200F-	GetMenuFlag	$2F0F-	InitPalette
$040F-	MenuVersion	$140F-	CountMItems	$220F-	GetMenuTitle	$300F-	EnableMItme
$050F-	MenuReset	$150F-	NewMenuBar	$230F-	MenuGlobal	$320F-	CheckMItem
$060F-	MenuStatus	$160F-	GetMHandle	$240F-	SetMItem	$330F-	SetMItemMark
$090F-	MenuKey	$170F-	SetBarColor	$250F-	GetMItem	$340F-	GetItemMark
$0A0F-	GetMenuBar	$180F-	GetBarColors	$260F-	SetMItemFlag	4350F-	SetMItemStyle
$0B0F-	MenuRefresh	$190F-	SetMTitleStart	$270F-	GetMItemFlag	$360F-	GetMItemStyle
$0C0F-	FlashMenuBar	$1A0F-	GetMTitleStart	$280F-	SetMItemBlink	$370F-	SetMenuID
$0D0F-	InsertMenu	$1B0F-	GetMenuMgrPort	$290F-	MenuNewRes	$380F-	SetMItemID
$0E0F-	DeleteMenu	$1C0F-	CalcMenuSize	$2A0F-	DrawMenuBar	$390F-	SetMenuBar
$0F0F-	InsertMItem	$1D0F-	SetMTitleWidth	$2B0F-	MenuSelect	$3A0F-	SetMItemName
$100F-	DeleteMItem	$1E0F-	GetMTitleWidth	$2C0F-	HiliteMenu	4310F-	DisaleMItem

TOOL #10 - CONTROL MANAGER

$0110-	Ct1BootInit	$0C10-	SetCt1Title	$1510-	TrackControl	$1E10-	GrowSize
$0210-	Ct1StartUp	$0D10-	GetCt1Title	$1610-	MoveControl	$1F10-	GetCt1DPage
$0310-	Ct1ShutDown	$0E10-	GrowSize	$1710-	DragControl	$2010-	SetCt1Action
$0410-	Ct1Version	$0F10-	ShowControl	$1810-	SetCt1Icons	$2110-	GetCt1Action
$0510-	Ct1Reset	$1010-	DrawControls	$1910-	SetCt1Value	$2210-	SetCt1RefCon
$0610-	Ct1Status	$1110-	HiLiteControl	$1A10-	GetCt1Value	$2310-	GetCt1RefCon
$0910-	NewControl	$1210-	Ct1NewRes	$1B10-	SetCt1Params	$2410-	EraseControl
$0A10-	DisposeControl	$1310-	FindControl	$1C10-	GetCt1Params	$2510-	DrawOneCtl
$0B10-	KillControl	$1410-	TestControl	$1D10-	DragRect		

TOOL #11 - SYSTEM LOADER

$0111-	LoaderInit	$0511-	LoaderReset	$0B11-	LoadSegNum	$0F11-	GetLoadSegInfo
$0211-	LoaderStartUp	$0611-	LoaderStatus	$0C11-	UnloadSegNum	$1111-	LGetPathName
$0311-	LoaderShutDown	$0911-	InitialLoad	$0D11-	LoadSegName	$1211-	UserShutDown
$0411-	LoaderVersion	$0A11-	Restart	$0E11-	UnloadSeg		

TOOL #12 - QUICKDRAW AUXILIARY

$0112-	QDAuxBootInit	$0312-	QDAuxShutDown	$0512-	QDAuxReset	$0912-	CopyPixels
$0212-	QDAuxStartUp	$0412-	QDAuxVersion	$0612-	QDAuxStatus	$0A12-	WaitCursor

TOOL #13 - PRINT MANAGER

$0113-	PMBootInit	$0A13-	PrValidate	$1113-	PrClosePage	$1B13-	LLDControl
$0213-	PMStartUp	$0B13-	PrStlDialog	$1213-	PrPicFile	$1C13-	LLDControl
$0313-	PMShutDown	$0C13-	PrJobDialog	$1413-	PrError	$1D13-	LLDText
$0413-	PMVersion	$0D13-	PrPixelMap	$1513-	PrSetError	$2313-	PrDriverVer
$0513-	PMReset	$0E13-	PrOpenDoc	$1613-	PrChoosePrinter	$2413-	PrPortVer
$0613-	PMStatus	$0F13-	PrCloseDoc	$1913-	LLDStartUp		
$0913-	PrDefault	$1013-	PrOpenPage	$1A13-	LLDShutDown		

TOOL #14 - LINE EDITOR

$0114-	LEBootInit	$0B14-	LESetText	$1314-	LECopy	$1B14-	LEScrapHandle
$0214-	LEStartUp	$0C14-	LEIdle	$1414-	LEPaste	$1C14-	LEGetScrapLen
$0314-	LEShutDown	$0D14-	LEClick	$1514-	LEDelete	$1D14-	LEScrapLen
$0414-	LEVersion	$0E14-	LESetSelect	$1614-	LEInsert	$1E14-	LESetHilite
$0514-	LEReset	$0F14-	LEActivate	$1714-	LEUpdate	$1F14-	LESetCarat
$0614-	LEStatus	$1014-	LEDeactivate	$1814-	LETextBox		
$0914-	LENew	$1114-	LEKey	$1914-	LEFromScrap		
$0A14-	LEDispose	$1214-	LECut	$1A14-	LEToScrap		

TOOL #15 - DIALOG MANAGER

$0115-	DialogBootInit	$1015-	IsDialogEvent	$1E15-	GetControlDItem	$2B15-	DetNextDItem
$0215-	DialogStartUp	$1115-	DialogSelect	$1F15-	GetItext	$2C15-	ModalDialog2
$0315-	DialogShutDown	$1215-	DlgCut	$2015-	SetIText	$2E15-	GetDItemValue
$0415-	DialogVersion	$1315-	DlgCopy	$2115-	SelectIText	$2F15-	SetDItemValue
$0515-	DialogReset	$1415-	DlgPaste	$2215-	HideDIItem	$3215-	GetNewModalDialog
$0615-	DialogStatus	$1515-	DlgDelete	$2315-	ShowDItem	$3315-	GetNewDItem
$0915-	ErrorSound	$1615-	DrawDialog	$2415-	FindDItem	$3415-	GetAlertStage
$0A15-	NewModalDialog	$1715-	Alert	$2515-	UpdateDialog	$3515-	ResetAlertStage
$0B15-	NewModelessDlog	$1815-	StopAlert	$2615-	GetDIItemType	$3615-	DefaultFilter
$0C15-	CloseDialog	$1915-	NoteAlert	$2715-	SetDItemType	$3715-	GetDefButton
$0D15-	NewDItem	$1A15-	CautionAlert	$2815-	GetDItemBox	$3815-	SetDefButton
$0E15-	RemoveDItem	$1B15-	ParamText	$2915-	SetDItemBox	$3915-	DisableDItem
$0F15-	ModalDialog	$1C15-	SetDAFont	$2A15-	GetFirstDItem	$3A15-	EnableDItem

Fig. 3 – Numeric order list of IIgs tools (part 3) . . .

similar to the *Vaporlock* that did appear in my *Enhance II.* An exact screen lock opens up all sorts of new video possibilities.

What kinds of new possibilities? Why don't you show me? For this month's contest, just tell me about some new, sneaky, or mind-blowing thing you can do with an exact or a partial software video lock. We'll have all of those usual *Incredible Secret Money Machine* book prizes for the best two dozen entries, and an all expense paid [FOB, Thatcher, AZ] *tinaja quest* for two for the very best entry of all.

One tip: be certain to defeat any and all interrupts during the time an exact screen lock is set up, or some very wierd results may happen.

What Technical Resources are Available for the Handicapped?

The personal computer has been the great equalizer for the handicapped, and we sure get lots of help line calls on this topic. Figure two lists many of the organizations and resources involving special education and therapy.

Both *Apple* and *Tandy* have quite excellent special education contacts and resources available, as do many local ham radio clubs.

As with just about any other field, the resources are there; all you have to do is dig into them on your own. All it takes is some time and effort.

Show me More of the Main IIgs Tool List

As we have seen over the past two months, a complete listing of all of the IIgs tools in numeric order is very hard to find. Yet, it is extremely useful for analyzing other people's working IIgs code. Our third installment appears in figure three.

The *tearing method* that lets you tear apart and analyze virtually any machine language code suprisingly fast and amazingly easy (I've even taught this to seventh graders.) does appear in my Enhancing Your Apple II, volume I. See you there.

Any New and Off-the-Wall Graphic Arts Resources?

Well, I do have a few loose ends that have been kicking around here for a while. Ferinstance . . .

```
%  Copyright c 1988 by Don Lancaster and Synergetics, Box 809, Thatcher,
%  AZ, 85552. (602) 428-4073  All commercial rights reserved. Personal use
%  permitted so long as this header remains intact. Demo disk costs $39.50.

%  Scans "raw" input text, converting to line-by-line strings for further
%  PostScript processing. Can be used for emulators that you can switch
%  into and out of under software command, for "\" character substitution,
%  to redefine the -escape- character or other special characters.

%  To begin scanning raw text, use the -startscan- command or an [esc]-g
%  alias. To stop scanning raw text, use an escape followed by your
%  -exitchar  character. Scanning also automatically stops on end of file.

%  For the maximum possible speed, code similar to this should be included
%  inside your actual justification routines. Otherwise, speed will be lost
%  if you end up scanning any particular character more than once. The
%  -stringwidth- command in PostScript is extremely slow.

6000 string /scanstr exch def

/exitchar 120 def       % use esc-x to exit scanner
/escsubchar 33 def    % use escape as is

/startscan {{ clear /more false def -1 { 1 add dup currentfile read not {pop
exit} if dup 92 eq {rslashproc}if dup 10 eq {pop /more true def exit} if

%  if you are not substituting escape, bypass the next line as a comment . . .

% dup escsubchar eq {pop 27} if

dup 27 eq {exitproc} if scanstr 3 1 roll put } loop scanstr exch 0 exch
getinterval

%  replace the following line with whatever proc is going to use the newly
%  created PostScript strings . . .

myjustproc

pop more not { exit} if} loop} bind def

%  The rslashproc routine does reverse slash substitutions to the raw
%  input text as if it was a PostScript string. Replace with /rslashproc
%  { } def if you do not want to use reverse slash substitutions.

/rslashproc {pop currentfile read not {pop /more false def exit} if
dup 41 eq { 1000 } if   % replicate right paren
dup 40 eq { 1000 } if   % replicate left paren
dup 92 eq { 1000 } if   % replicate reverse slash
dup 98 eq { pop 8 1000 } if   % substitute bs
dup 102 eq { pop 12 1000} if   % substitute formfeed
dup 116 eq { pop 9 1000} if   % substitute tab
dup 110 eq { pop /more true def exit} if   % end string on linefeed
dup 114 eq { pop /more true def exit} if   % end string on return
dup 48 ge { dup 55 le {dooctal} if} if
1000 eq { } {pop 1 sub dup 1 add 0 } ifelse   % substitute or replicate} def

/dooctal {48 sub  64 mul /oct exch def currentfile read not {pop /more false
def exit} if dup 48 ge { dup 55 le {dothird} {1000} ifelse} if} def

/dothird {48 sub 8 mul oct add /oct exch def currentfile read not {pop /more
false def exit} if dup 48 ge { dup 55 le {48 sub oct add} if 1000} if} def

/exitproc { pop currentfile read not {pop /more false def exit} if dup
exitchar eq {pop exit}{exch scanstr exch 27 put exch 1 add dup 3 -1
roll} ifelse} def

/[esc]g {startscan} def    % an alias for startscan

/[esc]x { } def                % reserve [esc]-x to exit
```

Fig. 4 – PostScript code for an input text scanner.

There is an outfit called *Photolabels* that makes photographic prints directly onto thin and self-stick photo paper, with prices starting at twenty cents each in small quantities.

This can be one sneaky way to add full color to all your present desktop publishing capabilities. You just peel and stick these onto whatever it is you are already printing. Several different sizes are available, ranging from "postage stamp" up to "postcard" formats.

Calendars are one obvious possibility, as are realtor listings. What others can you think of?

Actually, "real" color printing need not be all that expensive. The *Modern Litho* people will do all of the photography, typesetting, mechanical art, color seps, stripping and printing on a one-side, four color glossy sheet for a tad over seven cents each. In quantities of a few thousand or so. Longer runs are even cheaper.

As we've seen in previous columns, on-demand laser printing can be more than cost competitive with jiffy printing for smaller press runs, particularly if you do not know precisely how many copies you are going to sell, and especially if there are to be revisions and updates. Full details on this have appeared in previous *Ask The Guru* reprints, and you'll be hearing much more on *book on demand* printing in the future.

On the other hand, if you need a *known quantity* of modest quality user manuals or any other technical publications at a very low price, do check into those *Omnipress* folks. These people use paper plates and "shoot and go" litho techniques to hold down production costs. Their final quality is not all that awful, especially for tech manuals. The turnaround for fully bound books or manuals can be less than a week.

The GBC people have announced a new thermal binding scheme that resistance heats the actual glue in the binder, rather than needing a fancy "toaster" like the previous methods do. Most of their current binding tools and products are priced outrageously high, though, so don't get your hopes up on this one.

This one sounds sort of like those old hot dog cookers that directly electrocuted several hot dogs at once by connecting them across the ac power line. Sure enough, they got hot when you tried this. Totally inedible, but hot. They also both burned your tongue and left a bad taste in your mouth.

What is "Post-Justification Editing?"

It is probably the simpliest and most elegant way of dramatically upping the print quality of virtually any and all desktop publishing work. Post-justification editing is also extremely controversial since it kicks sand in the face of just about all of "them" at once.

Its only little advantage is that it does work like a champ. But nobody believes it until they try it. After that, they cannot live without it.

Anything you can possibly do to up the quality of any low resolution printing can help dramatically. And, conversely, anything that you do *not* do is sure to return to haunt you.

We've seen in the past how nearly everybody always insists on using the seventeenth lousiest gray with PostScript, and how this can be fixed with a few keystrokes. Hint: *106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen*. Or try a "135 25". Or "95 25". Or even "85 35". But *never* that putrid stock "53 45". Urp.

We have also seen how you can dramatically improve the legibility and appearance of typography as small as 3 points at 300 DPI, through a individual pixel by pixel "step and lock" technique.

In our upcoming October column, we will be seeing just how you can go to an upgraded and very sophisticated, three-stage progressive gonzo fill justify including such features as a minimum auto-kerning, hanging punctuation, drop caps, and lots of similar goodies.

But, let's assume that you are still using Ragemaker or something similar that has, at its best, some sadly mediocre fill justify routines. What can you do to improve your final text appearance?

Just this: *After* all of your final fill justified text is in its final form on the final page, you go back and let the original author *individually and by hand* adjust each and every one of the printed lines for the very best of possible aesthetics.

Paragraph five of corollary three of Swampfelder's seventh law of printing legibility states that any "pretty" text is much more readable and much more retainable than any "correct" text.

By purposely making all your text slightly wordy and purposely chosing less than the perfect word in each position in each sentence, you might dramatically improve both the reader enjoyment and comprehension – provided you do improve the "black balance" and minimize any "spaceticity" of the layout in the process, aiming for the smoothest and densest visual result.

So how do you go about doing post-justification editing?

Obviously, you start with a decent hyphenation routine. And then, you manually override that routine any time it doesn't do the best possible job. Many auto-hyphen routines will only break at the first possible place in a word, rather than at an optimum.

Next, you add or remove connectives. Nobody pays any attention at all to words like "a", "an", "the", "do", and/or "all". So, add or remove these to get the most uniform and the densest line.

Another ploy is to either use or not use contractions. Use of a "let's" takes up less space than does "let us"; and "aren't" will use less space than "are not", and so on.

A subtle way of improving the line aesthetics that most publishers do overlook is to use a slightly smaller type font for numbers or any capital strings. Fractional font sizes are a must for this sort of thing.

It also pays to stash a list of your favorite "alike but different somehow" words whose meanings are all more or less the same, but which will take up different amounts of space. "Some", "a few", and "several" are obvious examples.

Adjusting each paragraph so it ends in the middle of a line, rather than at the extreme end or beginning, can also greatly improve readability and appearance. Adding or removing words is the obvious way to handle this task.

You also want to avoid *widows* and *orphans*, which are single words or very short lines at the top or bottom of any column. These are especially bad at the beginning or end of any page. The same tricks you used to lengthen or shorten your paragraphs will also work here.

Using the widest feasable columns will also help. Columns less than "an alphabet and a half" wide will introduce all sorts of visual problems that are inordinately hard to overcome, even with many trips through your postjustification editing process. That really drove me up the wall back in volume one, before *Computer Shopper* went to their wider, and far more readable, columns.

Sometimes, working upside down can help you bunches, particularly when it comes to picking out shading problems, or flowing "rivers" caused by improper alignment of spaces on several successive text lines. Again, add, remove, lengthen, shorten, abbreviate, debreviate, or change the tense to eliminate any and all of this type of problem.

Another obvious use for all your post-justification editing is for fitting text to the available space.

And that is what post-justification editing is all about. With practice, it can be fast and simple, and will most often dramatically improve the appearance of your final images.

What is This Month's PostScript Utility?

We will have a heavy duty routine for advanced PostScript hackers this month that can solve a bunch of very sticky problems for you. The figure four code is for an input text scanner.

A text scanner can take a plain old text stream from any old source and convert it into strings, one for each line of text encountered. Many of the PostScript routines expect strings as input, and manually putting parenthesis around each text line can get old very fast.

So far, text scanning is no biggie. PostScript's *getline* command works just fine for this. And, most of those stock emulators are nothing but fancy text scanners in disguise.

But, suppose you want to write an emulator that you can switch into *or out of* at any time under your own full software control.

Or, suppose you want to make use of PostScript's "reverse slash" commands on "raw" input text files? The reverse slash commands normally will work only *within* internally defined strings.

Or, perhaps you want to handle such printing tasks as using several embedded "escape" sequences on a host that can only output printable low ASCII characters, or has other restrictions to non-printing codes.

Or access all of the 192 *Zapf* dingbats? Or do some sort of automatic character substitution?

Or maybe you want to read any old data base, and convert it into some proportionally spaced yet perfectly aligned columns in a mixture of fonts and different justify modes.

This new text scanner works by grabbing one input character at a time. Those characters are tested to see whether they are an end-of-line linefeed character, the substitution character, a reverse slash sequence, or an end-of-scan command.

If an end of line character, then the newly created PostScript line string is passed on to whatever it was that you wanted the string for in the first place. It is then further processed.

If a substitution character, then the needed code substitution is made. For instance, you might like to use a "!" in your host, instead of embedding escape commands, if your host does prohibit you from embedding any control commands into a PostScript or other textfile.

If a reverse slash, the next few characters get tested per the rules in figure five, and all the needed commands or characters are substituted into the line string. Finally, if an end-of-scan command, the scanner will kick out and drop you back into PostScript.

Scanning also ends automatically at the end of the file.

Two loops are involved. The inner loop builds a single text line on a character by character basis. The outer loop keeps building text lines so long as any additional characters remain, or until an end-of-scan command gets received.

Our upcoming new gonzo justify routines will make extensive use of this input scanner, so we'll be seeing lots more on this code.

Any input text scanner will, of course, slow you down. This particular one has been optimized for speed in several ways, but still stands some improvement. Ideally, when you finalize all of your PostScript code, you should combine input text scanning with some other tasks. In particular, it is unforgivable to have to scan each and every character two or more times.

Many more details on all this text scanning code, as well as ready-to-run versions of the latest updates on many of my previous PostScript goodies now appears in my new *Don Lancaster's PostScript Secrets*, available for most popular computers. Write or call for more info.

\b	–	substitute backspace
\f	–	substitute formfeed
\n	–	substitute linefeed
\r	–	substitute carriage return
\t	–	substitute tab
\(	–	substitute opening parenthesis
\)	–	substitute closing parenthesis
\\	–	substitute reverse slash
\nnn	–	substitute ASCII character of octal code #nnn For instance, \320 is an em dash.

Any other character following a reverse slash gets ignored. This is one convenient way of supressing any unwanted carriage returns forced by the formatting code in some word processors.

With the input text scanner of figure four, all of these reverse slash commands may be modified or extended in any manner.

Fig. 5 – PostScript "reverse slash" string substitution commands.

Don Lancaster's

ASK THE GURU

September, 1988

Sight reading an eexec file
Bad SX cartridge problems
Omnicrom and Kroy Kolor
Direct input from/to Apple
Cheap new fusion machine

I just got back from the AZ Apple Fiesta and assembly language developers conference. One big thing that struck me right off the top was how the style and flavor of all the leading Apple IIc, IIe and IIgs assembly tools exactly reflects the personalities and mind-sets of their creators.

While lots of useful results and otherwise good vibes came from the conference, the key problem of the totally unacceptable frustration level of the current IIgs programming environment never did seem to get itself properly addressed.

One most interesting sidelight of the conference was the tour of Bill Mensch's *Western Design Center*. Some of the finest and the most innovative new microprocessor designs in the world come from this family operation all done out of a remodeled home.

The dozen or so employees have time and again outperformed all of *Motorola* and *Intel* combined.

This is the home of the Apple IIgs CPU. Unlike some other companies I better not mention, WDC actually *uses* the products they design for all of their day-to-day operations. All the *Calma* development stations and the *Versatec* plotters are ultimately controlled by Apple IIgs computers.

Besides having some new 32-bit 65832 chips and some other unmentionable goodies in the works, WDC has a brand new in-circuit 65XXX emulator (ICE) board product that can leap tall buildings in a single bound.

While emphatically and positively not promoted as such, this ICE is the Apple software pirate's dream tool. It flat out defeats 100 percent of all known forms of all program software protection to date.

Few people realize that derivatives of the original 6502 chip are far and away the best selling microcomputers in the world today. The big players are *Sony*, who uses zillions of these in all their VCR's and whatever, the *Mitsubishi* folks with an extensive line of semi-custom devices (check out their super whiz bang M50734), the *Western Design Center*, with all their 65816 and newer products, *NCR* and *MOS Technology*, who are two of the commodity production houses, and *Commodore*, who now have a brand new 6502 in the works that does a 40 percent speedup by use of fewer machine cycles on such things as conditional branches.

Lots of helpline callers have asked when PostScript will be available for their Imagewriters and similar dot matrix printers. Well, it turns out there is a UNIX based and public domain British version of display PostScript now kicking around on the various bulletin boards. It's only a matter of time before someone adapts a display PostScript system like this one to one or more low end printers. Two products in the works are *GoScript* and *Freedom of the Press*.

I am in the process of chasing all these down. Stay tuned.

Let's see. Yes, we do continuously stock Adobe's red, blue, and green books here at *Synergetics*. These are absolutely essential to any intelligent use of PostScript. Start with the blue cookbook. While I have bunches of my own PostScript products available, my *Show and Tell* is usually your best starting point if you already have a PostScript printer, or my *Introduction to PostScript* video if you do not. Go for it.

I've also got a brochure or two for you that provides all sorts of insider sources for great free stuff. Write or call if you are interested.

As per always, this is your column and you can get tech help and off-the-wall networking by calling or writing me per the end box.

This month, we'll have a special one-time "double whammy" for all of you desktop publising fanatics – first details on a low cost machine to fuse Omnicolor or Kroy Kolor, and then info on how to "sight read" most any PostScript *eexec* file.

But first . . .

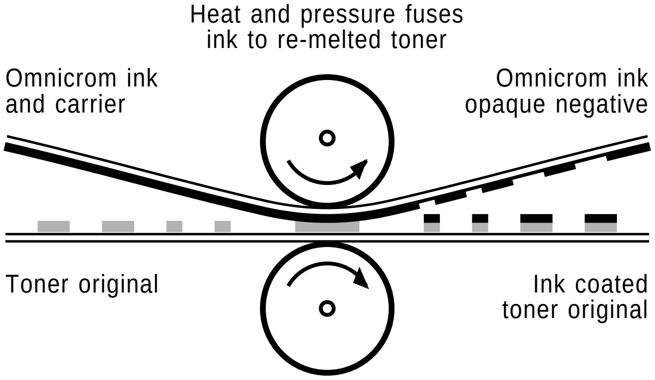

Heat and pressure fuses ink to re-melted toner
Omnicrom ink and carrier
Omnicrom ink opaque negative
Toner original
Ink coated toner original

Fig. 1 – How the Omnicrom / Kroy Kolor process works.

Does Anyone at Apple Ever Listen?

The Apple II Marketing Manager over at *Apple Computer* is Peter Sandys. Peter most definitely does welcome any of your courteous and well thought out letters on future Apple II directions; on any problems you have found with any existing products and systems; or on any suggestions for improvements of most any reasonable sort. I believe he is serious and will act on user input.

But, please, please do not overload him with personal tirades or ask him to referee a local dealer squabble. Peter's AppleLink mail address is SANDYS.

What is the Canon SX Cartridge Debacle?

The LaserWriter NT and NTX are both absolutely outstanding laser printers, but unless *Canon* immediately cleans up their act on the SX cartridges that go into them, both the NT and the NTX are history. Dead meat. Color them gone.

At present, there is as much as a 15:1 per-page toner cost penalty for using the newer LaserWriters, when compared to the original LaserWriter and the LaserWriter Plus. And, yes, all those older machines are just as black when they are on their second or third refill using a good third party toner source.

Which means that the NT and NTX are currently totally useless for any serious production work. I have been forced to go back to my older LaserWriters for all my book-on-demand printing, and for most of my other production work. Others seem to be doing the same.

As we've seen in several previous columns, you can get LaserWriter or LaserWriter Plus toner costs down into the 0.3 cents per page range. The NT and the NTX currently will often cost out at a nickel per page or even substantially higher.

More to the point, there is a 5:1 per page toner cost penalty compared to the competing *NEC LC890* user refillable PostScript laser printer that is currently running away with all of the marbles and then some.

Forgetting temporarily about the Canon SX cartridges being obscenely overpriced, my own experiences with these cartridges do reflect the many valid complaints I've been receiving.

Here's what is happening . . .

I bought five brand new *Canon* SX cartridges directly from Apple. The first cartridge ran out of toner after 1700 pages. The second cartridge developed two really awful scratches, long before it ran out of toner.

The third cartridge had a total mechanical jam when received and delivered *zero* copies. That fourth cartridge developed more serious scratches, again long before running out of toner. For some very strange reason, I am hesitant to so much as remove the fifth cartridge from its shipping box.

1 – Canon F21680 Fusion machine using
a FH1-0576-01 temperature pc card.

1 – 25K linear volume control
1 – 150K, 1/2 watt resistor
1 – Dialplate decal
1 – Dialplate decal overlay
1 – Push-on knob with pointer

Misc: 12 inches of red solid #22 hookup wire; 12 inches of similar green wire; 12 inches of similar black wire; 8 inches of solder; two 3/8 inch volume control nuts; 1 flat volume control washer; 1 internal tooth volume control lockwasher.

Note: One source of the Canon fusion machines, parts kits, and modified and tested units is Arlin Shepard at Lazer Products, 12741 E. Caley Avenue, Suite #130, Englewood CO, 80111. (303) 792 5277.

Fig. 2 – Parts list for a cheap Omnicrom fusion machine.

Canon is currently batting .000, at least in this inning. A trade to Baltimore is imminent.

I have asked around, and I am shocked to report that I have found no one using these cartridges that did not have their own equally bad horror stories to relate.

I would dearly love to show you how to refill these cartridges. But I have yet to receive one that was not so badly scratched that there was no point in refilling it. Others report a 33 percent one-time SX refillability rate.

Some refillers do report excellent refilling success if they immediately remove all of the toner from a factory fresh cartridge and replace it with a suitable third-party toner. They often will pass on the original toner on to their local diesel mechanic for use as valve grinding compound.

O.K., here's what has to be done and done now: The SX cartridges *must* be dramatically reduced in price and then absolutely *must* be made refillable by the end user. A toner that is far less abrasive *must* be used.

A drum that is far more scratch resistant *must* be used. A good drum conditioner and/or lubricant *must* be added. The poor engineering on the

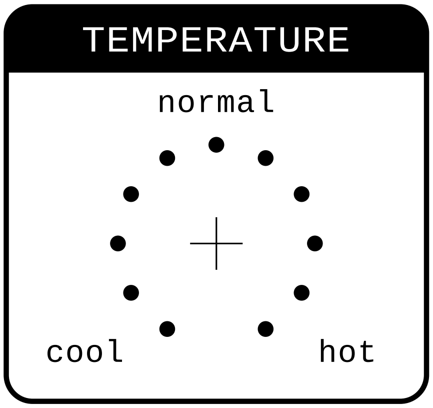


Fig. 3 – Full size temperature dial decal artwork.

drum wipers that cause scratching *must* be completely redone.

I sure would like to employ my fast and shiny new NTX for my production work here at *Synergetics*. But, I am instead forced into still using my 300,000 copy long-in-the-tooth LaserWriter Plus.

The obscene per-page toner costs and the scratchy images on the NTX simply are not acceptable.

Please continue to send along all your SX horror stories. More on this as the drama unfolds.

Tell me About Omnicrom and Kroy Kolor

For years, I've had a back-burner project going. The idea was to take an ordinary Xerox copy and run it through a magic machine where real ink would somehow stick only where the toner already existed.

Obvious uses would be to get truly dense blacks, to be able to provide "litho" quality images for my printed circuits, overheads, or for electronic artwork, to gain total color options, and to provide a durable raised ink thermography process, for letter-heads, for custom business cards and even for use when printing in Braille.

It turned out that an English outfit by the name of *Omnicrom* beat me to the punch. As figure one now shows us, Omnicrom reasoned that toner was really a mixture of black stuff and hot glue. You could think of a copy as a piece of paper that had hot glue selectively placed only where you really wanted it.

By putting a dry ink based carrier in contact with the Xerox copy and applying heat and pressure, the toner would remelt and grab the dry ink off the carrier sheet.

Presto. The near instant conversion of a copier or a laser printed output into brilliant metallics, bright mattes, a solid black, silvers, golds, and even some pearl effects.

Along with a unique shiny high gloss varnish or even a true plastic laminating for scuff-resistant menus or book covers.

As a bonus, the carrier sheet can also become an "instant negative", useful for such things as overhead transparencies. If you wanted to, you could even reuse any remaining part of any carrier as often as you liked.

But there were serious problems. The Omnicrom people were Brits and thus predictably and monumentally mismanaged all of their marketing efforts in the colonies. Their early materials were not all that reliable. Worst of all, they saw nothing unconscionable about the charging of over $1400 for a fusion machine that was nothing but a pair of heated rollers that would turn for you.

A few years ago, the *Kroy Kolor* people did become an Omnicrom licensee and then dramatically improved all the materials, added more colors and options, and made them much easier to get. They also did improve and modestly lower the cost of all their fusion machines.

You can get several free evaluation

( ) Verify that the unit to be modified is in fact a Canon F21680 machine and then unplug the line cord from the AC outlet.

( ) Remove the center lid by raising the green handle and removing the two black screws at the rear and the two silver screws on the inside. Be gentle!

( ) Remove the right cover by removing the two silver screws at the rear, the black screw at the right bottom, and the black screw at the front bottom. Again, be gentle!

( ) Cover the temperature decal with a similar sized clear self-stick overlay. Neatly trim the decal to just outside the black border.

( ) Clean the front of the right cover and apply the temperature decal so it is 3/8 of an inch in in and 3/8 of an inch up from the lower lefthand corner. (See Figure five)

( ) Drill a 13/32 inch hole at the crosshairs on the temperature decal. This is easiest to do with a Vise Grip UNIBIT, but a pilot hole and reamer can be used instead. Deburr the hole.

( ) Mount the volume control in the hole. Use a nut and a lockwasher behind the panel and a nut and flat washer in front of the panel, adjusting for a flush mounting.

( ) Solder the 150K resistor to the center lug of the volume control, keeping the lead both reasonably short and mechanically strong. (See Figure six)

( ) Form a loop in the free end of the resistor and trim the lead. Strip 1/4 inch off of both ends of the green wire and solder one end to this resistor loop. (See Figure six)

( ) Strip 1/4 inch off both ends of the black wire and solder it to the leftmost lug of the volume control when viewed from the rear bottom. (See Figure six)

( ) Strip 1/4 inch off both ends of the red wire and solder it to the rightmost lug of the volume control when viewed from the rear bottom. (See Figure six)

( ) Twist all three wires tightly together for all but the last inch of their length.

( ) Unplug the black and white motor connector from the innermost circuit board. Unplug the two heavy red wires from the motor speed sensing power resistor.

( ) Remove the three large Phillips screws that hold the circuit boards in place. You may have to restrain the white circuit board spacers to keep them from turning when you do this.

( ) Slide the two circuit boards out where you can work on them. You may have to work some slack in the yellow thermistor wires. Unplug the five pin black and blue connector.

( ) Verify that the yellow thermistor wires go to a FH1-0576-01 printed circuit board.

( ) Route the twisted black, red, and green wire between the motor starting capacitor and the motor control power triac. Then route it along the yellow wire, through the small hole, and to the FH1-0576-01 printed circuit board.

( ) Loop and solder the black wire to the inside end of diode D303. (See Figure six)

( ) Loop and solder the green wire to the outside end of resistor R307.

( ) Loop and solder the red wire to the outside end of diode D304.

( ) Replace both circuit boards, the three large Phillips screws, the black and white motor connector, the two red power resistor wires, and the blue 5 pin connector. Redress the yellow thermistor leads back the way they were.

( ) Verify that the right cover can go back on without pinching any wires and that there are no wires anywhere near the large gear or other moving parts.

( ) Verify that all connectors and wires are now secure.

( ) Replace the right cover and the lid using the original screws.

( ) Press the knob onto the volume control shaft, being certain that the pointer points straight up to NORMAL when at the center of its rotation range.

( ) Refer to the text for the checkout procedure.

Fig. 4 – How to make the fusion machine modifications.

samples by calling Randy Bailey at Kroy. They also now have lots of sign-building accessory kits and some heavier gloss printing stocks.

But, judging from the *Perrier* in all of their corporate birdbaths, Kroy does appear to be a company of, by, and for Yuppies. "Low end user cost" is not presently in their vocabulary. If you do not own a BMW, they do not appear to want you as a customer.

So, it might still take a long while before some genuine competition can drive the end user costs down to the nickel per sheet mass market range where they rightfully belong.

Kroy Kolor is a fantastically great product wherever its sixty cents or so per sheet cost can be justified. This product totally revolutionizes what you can do with a copier or a laser printer. The sad thing is that it could do so much more if only it were sanely priced for the end user.

Show me how to Build a Cheap Omnicrom Fuser

You do not need $1400 to use either Omnicrom or Kroy Kolor. All you need is heat and pressure.

For instance, a plain old iron and a muslin pressing cloth will do the job just fine. You can also use those existing fusion rollers inside your laser printer or copier, by making a second pass while hand feeding a blank page. The process works best with the metallic colors. But, you might occassionally get a misfeed, wrinkles, or dropouts.

A few of the newest copiers are set up to directly use Kroy Kolor.

One trick that works well and can dramatically drop the price is to use spot color. Ferinstance, on a letterhead, you tape a small piece of Kroy Kolor applied only to the logo and then run it back through the printer. Be sure to use a very low tack tape, such as the Scotch *Post-It Cover Up Tape*, from your local office supply.

But there is a better way. It turns out there is a wondrously bizarre machine that is available today on the surplus market known as a *F21680 Canon Fuser Unit*.

Nobody (and especially all of the Canon dealers!) had even the slightest idea what these machines were for or how to use them, so they were all dumped at fire sale prices.

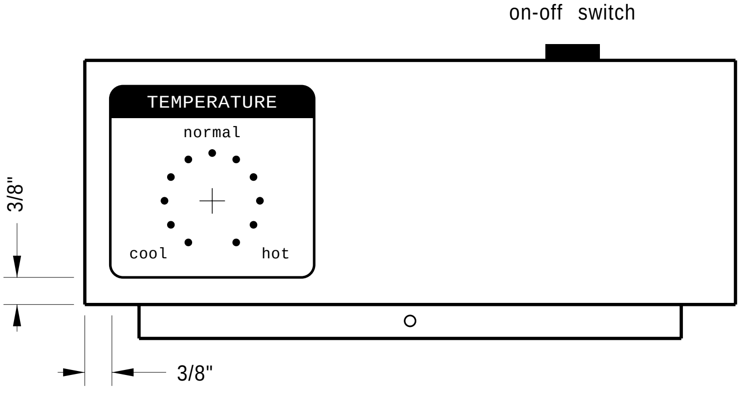


Fig. 5 – Location of the new fusion dial decal.

Rumor has it the machines were somehow involved in creating color overhead transparencies.

At any rate, The only difference between a genuine Omnicrom fusion machine and a Canon fuser unit is around 5:1 in cost and the fact that the stock Canon fuser machine was preset to a fixed lower temperature.

Fortunately, for sixty cents worth of parts and twenty minutes work, you can convert the Canon unit into a beast that actually will outperform the Omnicrom machine and do so at a tiny fraction of the going cost.

That super rugged Canon unit does give better results both because of a slower feeding speed and several self-cleaning roller wiper pads. It works especially well with the SX toner cartridges, such as those used in a LaserWriter NT or NTX.

The only little problem I have found with the modified unit is that you have to trim your laminating film to a maximum width of 8-1/2 inches. Kroy's *Color Plus* machine shares the same problem, at least at present.

Figure two shows you a parts list for this mod. Figure three is a full size replica of the new dial decal. Figure four gives you the detailed instructions, while figure five shows you where to position the decal on the front of the machine. Finally, figure six is a pictorial for the mod.

The fusion unit does its thing by comparing a thermistor's resistance

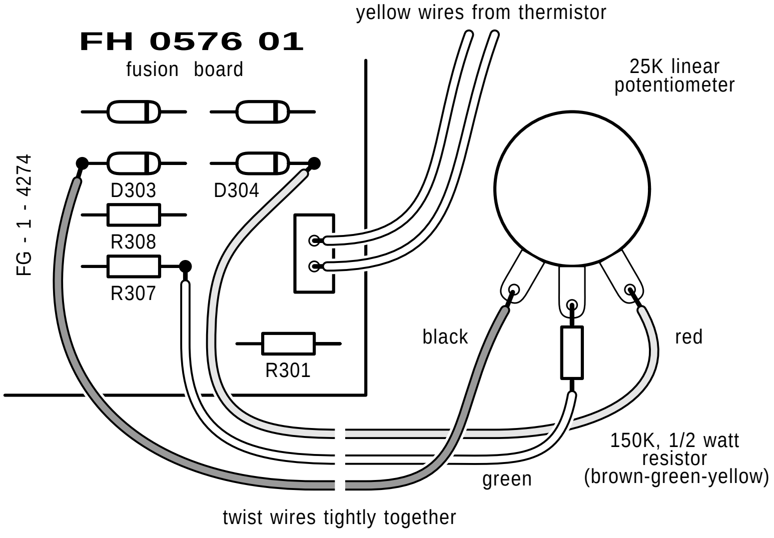

FH 0576 01
fusion board
FG - 1 - 4274
D303
D304
R308
R307
R301
yellow wires from thermistor
25K linear potentiometer
black
red
150K, 1/2 watt resistor (brown-green-yellow)
green
twist wires tightly together

Fig. 6 – Pictorial of the Canon Fuser Machine Modification.

against a fixed voltage reference. The modification lets you raise or lower that internal reference by sourcing or sinking extra current.

The temperature range is set by a new 150K resistor. A higher value *restricts* the range, while a lower value will *extend* it. A half-watt resistor is chosen here because it is physically stronger than a quarter watt one.

The checkout procedure is fairly simple. Center your new temperature control and then apply the power. The panel LED should start blinking a dim green and the internal fusion lamp should come on. After one minute, the fusion lamp should go out and the LED should change to a continuous green.

Advance the temperature control clockwise by one quarter of a turn. The fusion lamp should come on for three or four seconds.

Retard the temperature control fully counterclockwise and then wait a few minutes. Eventually the fusion lamp should come back on for a few seconds. When the lamp goes back off, then advance the control by one quarter turn. The fusion lamp should go back on again for a few seconds.

If all of these tests are passed, center the temperature control and try a metallic Kroy Kolor sheet. Use the "normal" setting for metallic foils, a somewhat higher setting for the matte colors, and a slightly lower setting for the laminating film.

Some users report better feeding with the exit rollers depowered. Try both powered and unpowered and see which you like better.

Here's another tip: If you run a toner copy through the machine in contact with a thin sheet of polyester "mylar" film or by using an "empty" Kroy Kolor carrier, your copy will *Bakerize*, giving you a more durable, blacker, and a semi-gloss finish. This is handy for such things as quick and dirty business cards, and is a zero cost process.

Among many other sources, these surplus Canon fuser units are now available by way of Arlin Shepard at *Lazer Products*.

Modification kits and some fully modified and tested units are also available, all at a tiny fraction of the current cost of the real Omnicrom fusion machines.

What is This Month's PostScript Utility?

Many downloadable fonts and lots of other PostScript utility programs are stored in an *eexec* data format. As you might expect, far and away the number one PostScript user demand on both the helpline and on our PostScript BBS is for a simple and easy way to read and understand all of the eexec files.

First, set up the following . . .

(1) A two-way communications environment that lets you receive and record error messages.

(2) A method of transmitting a PostScript file up to a cursed character, followed by a [d].

(3) An error trapper that dumps the stack, such as on page 205 of the green book.

Then, try these two tools . . .

(4) The CHOPPER works by truncating the eexec file and seeing what kind of error messages you get back. From the stack dump, the eexec file can be reconstructed.

(5) The INSERTER works by adding an extra FF or AA into the eexec file and seeing what kind of error messages you get back. From the stack dump, the eexec file can be reconstructed.

Here's a practice file to get you started . . .

```
currentfile eexec
3a6407b312f073e1f86dc6433724cca95ce3654946cebf2cf38791da40c2cc45
1d47ce34c2f6c0f1f6a2032c50a968def6c962731e815302344a7c3919d99ce1
14b84aab08fbed4e7a0a2af900f9e86efd82231015e6ea6a219d87e863cb56e2
153618cf3544d1a1217acba9d90343d8da484f02fa5b59365c94b678044f9098

                ----userdict/RD{string currentf
                ile exch readstring pop executeo
                nly}put userdict/ND{noaccess def
                 executeonly}put dup/Private 8 d
```

Fig. 7 – How to "sight read" a PostScript eexec file.

With some patience, eexec files can be directly "sight read", without needing anything fancy in the way of insider knowledge or exotic tools. Everything you need is right there in the red and blue and green books.

There are compelling and overwhelming advantages to converting all of your downloadable fonts into ordinary textfiles. Once converted, they often will store in less than half the space and download in less than half the time. You can now create "short fonts" where you only need a few special characters for an often used logo or label.

You can also now add or modify any character anywhere in the font while rearranging the scenery to suit yourself. Or extract character shapes for special uses. Such things as the complement bars needed for electronic text are now easily done. You can also now do the true perspective, "3-D" and other specialized font transformations much faster than you could by using the slow pixel line remapping that was demanded by eexec format files.

At any rate, figure seven shows you how to sight read an eexec file. What you do is get into a two-way communicating environment using an error trapper that can dump the stack dump.

You then use either a *chopper* tool that downloads the file up to a cursed character, followed by a [d] end-of-file; or else an *inserter* tool that adds an extra FF or an AA into the eexec file hex pair data stream.

With either tool, the error trapper's error message and stack dump will return enough information that most portions of most eexec files can now readily be reconstructed.

There are advantages and limitations to both the chopper and the inserter. There are also advantages and limitations to working forward or backward through the eexec file. I will let you check out and explore all of these on your own.

Figure seven also does give you a sample fragment of an eexec file and its standard text equivalent. Use this for practice before you begin.

More powerful tools can easily be added to speed up and automate the eexec sight reading process.

Don Lancaster's

ASK THE GURU

October, 1988

Duplex color tricks
Gonzo Justify Stuff
Color proofing systems
Anti-Scratch Pixie Dust
New IIgs System Software

Boy, a whole flock of them flew over that time. Apple has at long last officially flushed Pascal. Pascal was basically a laboratory animal that had escaped. Unfortunately, it did do an incredible and incalculable amount of damage to both academia and general computer literacy well before its long overdue extermination.

The heir apparent would seem to be the "C" language, helped along by some closely linked and tightly written assembly language modules.

The APW development system on the IIgs and the MPW development system on the Mac are good places to start using "C". Both of these are now available through APDA.

What about UNIX or A/UX? While UNIX is written in "C", I personally feel that (a) It is insanely bloated, (b) It is already hopelessly obsolete, and (c) anything that AT&T is in favor of has got to be suspect.

Let's see. There is a new version 3.2 system master disk for the IIgs. More on this shortly. The bottom line is that their printer driver improvements are far too little done far too late, but that there's lots of other neat stuff included in the upgrade.

Quite a few of you helpline callers have been asking for methods to run PostScript on the low-end dot matrix printers. Well, to do so would be the equivalent of putting a new *Porsche* engine onto a skateboard.

Nonetheless, the *LaserGo* people are the first ones out with such a product. Right now, it is mainly for IBM use, but Apple versions are certain to follow.

Apple Computer does have a few special interest publications that you might want to be aware of. For you developers, there is their *Viewpoints* newsletter and their *Outside Apple* quarterly magazine.

For all of you bureaucrats, check into *Innovations*. For librarians, there is the *Apple Library User's Group*, and, among their many educational publications, *Wheels for the Mind* is just about the best.

Meanwhile, *Adobe Systems* has a free *Font and Function* typography catalog out that includes some useful design and layout ideas.

Turning to all my own products, besides the *Ask the Guru* reprints, we now also have the *Hardware Hacker* reprints from the sister column to this one over in *Radio-Electronics*.

There's a new disk for the gonzo justify stuff we'll feature here, and, as usual, our *PostScript Show and Tell*, available for most computers.

The gonzo biggie this month is just that – the gonzo justify I have been working on for several years now.

But first . . .

What is in the new IIgs v3.2 System Master Disk?

Well, let's see. First, it boots faster and is better organized, and literally hundreds of bugs have newly been corrected. Or at least changed.

ProDOS 16 can now boot over AppleTalk, and the bugs in handling very large and single byte files have been corrected. The random memory trashing has also been fixed, along with an easing of all the SCSI hard disk format problems.

The printer drivers have been improved somewhat in a too little and too late manner. The *ImageWriter* driver is now faster on images and supports vertical condensed printing. A new and totally useless *ImageWriter LQ* AppleTalk driver will now recognize the machine but will not support any of its features.

While there are substantial new improvements in those LaserWriter drivers for the applications willing to play Apple games in Apple's way, a plain old *Send PS* is still conspicuously absent. You can, however, still rearrange IWEM to make it into an acceptable no-nonsense PostScript driver for custom uses.

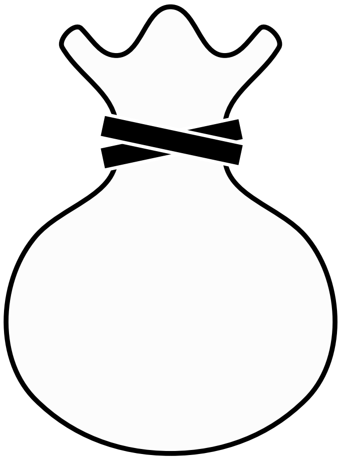
Also still missing are the crucially needed drivers for Epson and other dot matrix printers, for LaserJets, and for ink jet printers.

There are several new tool sets, including one for audio compression and expansion. These can reduce file sizes by fixed factors of 2:1 or 8:3.

The window manager has been greatly improved and includes a new ap-note with details on defining your own windows.

There is also an inch thick bunch of new documentation, including full specs for the Sound Tools, the Note Synthesizer, Audio expander, MIDI tools, and the Note Sequencer.

For details on the upgrade, see your local Apple dealer or your user



Make from the toe of
a child's athletic sock,
a rubber band, and two
heaping teaspoons
of pixie dust.

Fig. 1 – A pixie dust duster.

```
%  GONZO JUSTIFY MODULES  gonzo.dl.2    version 100.4    July 28,1988

%  Copyright c 1988 by Don Lancaster and SYNERGETICS, Box 809, Thatcher, AZ, 85552
%  (602) 428-4073. All commercial rights reserved. Personal use permitted so long as this
%  header remains present and intact. $39.95 Disks are available for all major computers.

/persist true def  persist {serverdict begin 0 exitserver} if /sfix 0 def /cfix 0 def
/justx (justF) def  % justification l = left f = full, etc.
/endtheline { closeline lastparline {pop pop}{startnewline}ifelse} def /closeline {/fontsave fontn 4
get def printsubstrings fontsave changefont /ypos ypos yinc sub def colcheck} def

% swallowandhang removes any trailing spaces from the line where they would interfere with
% fill and right justify. It then optionally hangs the punctuation as needed.

/hangflag true def /str ( ) def /hanglist (-.,:;"')\261\320) def /hangfract 0.67 def /reallyhang {
hanglist { hangchar eq  {str 0 hangchar put str stringwidth pop hangfract mul sparechange exch
add /sparechange exch def exit }if } forall} def /hangpunct {dup dup /hangchar exch def 97 lt
exch 122 gt or {reallyhang} if exit} def

/swallowandhang {lastchar 1 sub  -1 0 {linestring exch get dup 32 eq {pop /#spaces #spaces 1
sub def /#chars #chars 1 sub def /sparechange sparechange spacewidth add def}{hangpunct}
ifelse} for} def

/justF {swallowandhang #spaces dup 0 gt {lastparline {sparechange txtwide 0.10 mul ge {/sfix
maxsstretch 0.40 mul def}{/sfix 0 def} ifelse /cfix 0 def}{sparechange exch div /scfix exch def
xpos1 scfix dup maxsstretch lt {/sfix exch def /cfix 0 def}{pop maxsstretch /sfix exch def scfix
maxsstretch sub #spaces mul #chars 1 sub div dup maxcstretch lt {/cfix exch def}{maxcstretch
dup /cfix exch def sub #chars 1 sub mul #spaces div sfix add /sfix exch def} ifelse
}ifelse}ifelse}{pop /sfix 0 def /cfix 0 def} ifelse endtheline} def

/justL {/sfix 0 def /cfix 0 def endtheline} def

/justR {/sfix 0 def /cfix 0 def swallowandhang gsave sparechange 0 translate endtheline
grestore} def

/justC {/sfix 0 def /cfix 0 def swallowandhang gsave sparechange 2 div 0 translate endtheline
grestore} def

/changefont {dup fontn 4 get ne {fontn exch 4 exch put fontn cvx exec /spacewidth ( )
stringwidth pop spacestretch add charstretch add def}{pop} ifelse} def

/changejust {dup justx 4 get ne {justx exch 4 exch put}{pop} ifelse} def

/escapeproc {pop linestring 1 index /sparechange exch def 2 index 1 add dup 1 sub /oktohere
exch def get /cmnd# exch def createss /ssok false def

% cmnd#

( ) dup 0 cmnd# put cvn gonzolink exch get exec exch 1 add dup 1 add /ssstart exch def
/oktohere {ssstart} def exch exit} def

/otherproc {{dup (\033) eq {escapeproc} if dup (\n) eq {pop dup /sparechange exch def 1 index
/oktohere exch def createss justx cvx exec exit}{pop exit} ifelse}loop } def

/spaceproc {/#spaces #spaces 1 add def pop dup /sparechange exch def 1 index /oktohere
exch def /ssok true def spacewidth sub dup 0 lt {createss justx cvx exec}{dup /sparechange
exch def 1 index 1 add /oktohere exch def}ifelse} def

/commandscan {dup ( ) eq {spaceproc}{otherproc} ifelse} def /fontn (font?) def

% during scan, the running line sum is on the stack above the running character counter.

/firstuseflag true def % removes PostScript stringsave bug

/gonzojustify {firstuseflag {fontn 4 50 put 49 changefont /firstuseflag false def} if dup length 1 lt
{pop (  )} if dup /linestring exch def length 1 sub /lslen exch def /lastparline false
def/sparechange 0 def startfirstline {linestring 2 index 1 getinterval dup ( ) gt {stringwidth pop
sub charstretch sub dup 0 lt {#spaces 0 eq {1 index /oktohere exch def /ssok true def} if ssok
{createss} if justx cvx exec} if} {dup ( ) eq {spaceproc} {otherproc} ifelse} ifelse exch 1 add dup
lslen gt {exit} {exch} ifelse} loop /lastparline true def /ssok true def dup /oktohere exch def exch
dup /sparechange exch def ssok {createss} if justx cvx exec /ypos ypos yparendadj sub def} def
/yparendadj 0 def

% fillsubstrings - routines to create and fill the ssarray

/ssarray 25 array bind def % creates the array at load time
/colcheck {} def % default bottom of page check

/startnewline { /#ssa 0 def /xpos1 xpos indentcount 0 gt {pm add} if def /shiftproc {[xpos1 ypos
(moveto)] true} def /ssok true def /#spaces 0 def /#chars 0 def {linestring oktohere get dup 10
eq {pop 32} if 32 eq {/oktohere oktohere 1 add dup lslen ge {1 sub } if def}{exit} ifelse} loop
/ssstart oktohere def pop pop ssstart 1 sub txtwide indentcount 0 gt {pm sub /indentcount
indentcount 1 sub def} if} def

/startfirstline {/#ssa 0 def /xpos1 xpos pm add def /ssstart 0 def /ssok true def /shiftproc
{[xpos1 ypos (moveto)] true} def /#spaces 0 def /#chars 0 def ssstart txtwide pm sub} bind def
/newshiftproc {/shiftproc false def} def /indentcount 0 def

/createss { oktohere dup /lastchar exch def ssstart sub dup /#chars exch #chars add def /sslen
exch def ssarray #ssa mark spacestretch charstretch ssstart sslen shiftproc fontn 4 get] put
/#ssa #ssa 1 add def newshiftproc} def                                        ( more . . . )
```

Fig. 2A – My PostScript gonzo justify routines . . .

group. Or check on AppleLink. As a reminder, a free list of all your local user groups is available by calling up (800) 538-9696, extension 500.

What are Color Proofing Systems?

A *color proofing system* is any way of creating an accurate mockup or a prototype before an expensive and high volume color printing job is begun. There are dozens of different color proofing systems available.

At least in theory, some of these should be ideal for newer desktop publishing and LaserWriter uses.

Unfortunately, it is extremely difficult to get *any* information on any of these, so I'll let you tell me, rather than vice versa.

We already know about the *Kroy Kolor* process that uses a carrier to convert black toner images into most any metallic or matte color.

And, we have briefly looked at the *Color-Key* and the *Scotchcal Label* processes from *3-M*. These use a simple photographic chemistry to give you clear and colored areas on a polyester sheet or label stock.

The *LetraSet* people have some stuff called *LetraChrome* proofing materials, but these turkeys were so snotty to me that I hesitate to even mention their products at all.

Then there is *IdentiColor*, who does seem to be at the center of the color proofing universe. They also can do decals and metal foils. What they do when you write or call is send you a big packet of absolutely beautiful and mind-blowing samples, along with some utterly confusing and totally undecipherable ads.

Then, for a mere $5000 extra, they offer to send you a salesman who will explain the incredibly confusing ad materials to you.

There are also at least two mystery processes called *Chromatec* and *MatroColor* that I now know absolutely nothing about. And I suspect there are at least a dozen more.

And, in the weaving stores, of all places, are shelves of little colored photo-emulsion bottles that you can use to stick LaserWriter images onto nearly anything.

Obviously, there are great heaping bunches of off-the-wall opportunities here that are going begging.

So, let’s make a contest out of it. Tell me everything you know about any color proofing system. There will be several *Incredible Secret Money Machine* books for just about any entry at all, and an all-expense paid (FOB Thatcher, AZ) *tinaja quest* for two for the very best.

How can I Fake Duplex Color?

Duplex Color is a sneaky way of brightening up a printed page without going to expensive and registration-critical full process color. What you do is make two passes through the press. The first time, you put down large squares or highlight blocks of color. On the second pass, you put down your black text and graphics.

If you quickly flip through your *Computer Shopper*, you’ll find lots of examples of duplex color use.

I have found a good way to fake duplex color on the LaserWriter or on any other laser printer or Xerox copier. While it is not cheap and is not suited to high volume uses, the results can be quite spectacular.

Here’s what you do: Say you want black words inside a blue box. First, you make a black toner copy of all the words. Next, make yourself a black toner copy of the box.

Now, *Kroy Kolor* or *Omnicolor* the words. Then you *throw this printed sheet away and save the carrier*.

Now, look carefully at your color carrier. You’ll now have clear where the words were and blue elsewhere on the carrier.

Finally, you reuse this "negative" carrier sheet over the black box. The blue goes everywhere the words were *not*, and the black shows through.

Instead of that usual printer’s route of putting the color down first and then overlaying the black text, you put the color *on top*, letting the black show through *only* where you want words or border art.

Sneaky, Huh?

Here’s two additional Kroy Kolor tips. In case you missed it, last month we showed you several very cheap alternatives to those overpriced Kroy Kolor fusion machines.

Also, if you take an "empty" Kroy Kolor carrier, or else just a plain old half mil sheet of overlay clear mylar or polyester, and then run it and a black toner copy through an iron, a

```
% printsubstrings - prints or compiles a previously created ssarray array of form [[string1]
% [string2]...[stringn]] and length #ssa. Each substring array is in the form [-spacestretch-
% -charstretch- -wstringstartpointer- -wstringlengthpointer- -[moveproc]- -mproctrue- -font-].
% If compile is true, a compile returns values to the host. If compile is false, the array
% is imaged in the normal manner.

% logic and format for currentpoint shift:  -false-  use existing currentpoint;
% [6 3 (rmoveto)] -true- execute currentpoint proc; [1 2 (moveto)]  -true- execute currentpoint
proc

/compile false def /stalltime 20 def % compile time delay for host (optional)
/stall {stalltime {37 sin pop} repeat} def

% warning: compile not yet fully debugged -- needs to recognize shifts for rj and cj

/printsubstrings{ compile {0 1  #ssa 1 sub {ssarray exch get aload pop dup 0 ne {fontn exch 4
exch put (\r) print flush stall fontn print flush stall (\r) print flush stall }{pop} ifelse {{40 string cvs
print ( ) print flush stall} forall (\r) print flush stall} if linestring 3 1 roll getinterval /schold exch
def exch 8 string cvs print flush stall ( 0 32 ) print flush stall 8 string cvs print flush stall ( 0 ()
print flush stall schold print flush stall () awidthshow\r) print flush stall} for (\r) print flush}

{0 1 #ssa 1 sub {ssarray exch get aload pop dup 0 ne {dup fontn 4 get ne {fontn exch 4 exch
put fontn cvx exec }{pop} ifelse}{pop} ifelse {aload pop cvx exec} if linestring 3 1 roll getinterval
3 1 roll cfix add 3 1 roll sfix add 3 1 roll
0 exch 0 4 1 roll 32 4 1 roll awidthshow} for } ifelse} bind def

%  The gonzolink dictionary links the one-character escape commands to their gonzoprocs.
%  While this is completely flexible, I recommend using numerics for font selections,
%  capital numerics for tabs; uppercase for mode and justify selections; lowercase for
%  immediately executed PostScript utilities and value changes; and high ASCII for anything
%  that is special for one specific or unusual job.

/gonzolink 255 dict def gonzolink begin

/0 {48 changefont} def  /1 {49 changefont} def  /2 {50 changefont} def  /3 {51 changefont} def
/4 {52 changefont} def  /5 {53 changefont} def  /6 {54 changefont} def  /7 {55 changefont} def
/8 {56 changefont} def  /9 {57 changefont} def  /- {45 changefont} def  /= {61 changefont} def

/C {67 changejust} def % center justify
/F {70 changejust} def % fill justify
/L {76 changejust} def % left justify
/R {82 changejust} def % right justify

/a {amacro} def /b {bmacro} def /c {cmacro} def /d {dmacro} def /e {emacro} def /f {fmacro} def

/g {} def % to prevent reentry
/h {/ypos ypos yinc 2 div add def} def
/i {initialcap} def  % initialcap
/l {/oktohere oktohere 1 add def endtheline} def % force new line
/n {nobreak} % conditional formfeed
/p {/pm pmnorm def} def % normal pmargin
/s {showpage} def % showpage
/x {} % exit should be trapped by scanner
/y {/ypos ypos yinc add def} def % negative linefeed
/z {/pm 0 def} def      % zero pmargin
end

/nobreak {ypos yinc 6 mul sub ybot lt {/ypos ybot def} if} def
/initialcap {/ypos ypos yinc add yparendadj add def /pm dropindent def /indentcount dropcount
def} def

%  stringmacro lets you do a series of gonzolink commands with a single macro keystroke. For
%  instance, /amacro {(z3c) stringmacro} def picks a centered font3 with no paragraph for
%  an embedded [esc]-a.

/smac ( ) def /stringmacro {{smac exch 0 exch put smac cvn gonzolink exch get cvx exec}
forall} def () cvn {startgonzo} def

% callout justify routines

%  These are used for "quick and dirty" layouts and for illustration callouts. No scanner
%  is used. cc and cr align themselves on xpos, independent of txwide.

%   -xpos- -ypos- (String message) cl --> left justify
%   -xpos- -ypos- (String message) cf --> fill justify
%   -xpos- -ypos- (String message) cc --> center justify
%   -xpos- -ypos- (String message) cr --> right justify

/cl {save /snapcl exch def /msg exch def /ypos exch def /xpos exch def /justx (justL) def msg
gonzojustify snapcl restore} def

/cf {save /snapcf exch def /msg exch def /ypos exch def /xpos exch def /justx (justF) def msg
gonzojustify snapcf restore} def

/cc {save /snapcc exch def /msg exch def /ypos exch def 2500 sub /xpos exch def /txtwide 5000
def /justx (justC) def msg gonzojustify snapcc restore} def

/cr {save /snapcr exch def /msg exch def /ypos exch def 5000 sub /xpos exch def /txtwide 5000
def /justx (justR) def msg gonzojustify snapcr restore} def                    ( more . . . )
```

Fig. 2A – My PostScript gonzo justify routines . . .

```
/cr {save /snapcr exch def /msg exch def /ypos exch def 5000 sub /xpos exch def /txtwide 5000
def /justx (justR) def msg gonzojustify snapcr restore} def

% end of gonzo justify.  begin scan converter

% The gonzo justify routines expect input as () delimited
% strings and use the -escape- character for commands.

% An input scan converter can be used for " character substitution, to redefine the
%  -escape- character or to solve system problems.

6000 string /scanstr exch def /exitchar 120 def % for esc-x loop exit
/escsubchar 33 def % for use escape as is

/startgonzo {{ clear /more false def -1 { 1 add dup currentfile read not {pop exit} if dup 92 eq
{rslashproc}if dup 10 eq {pop /more true def exit} if
% dup escsubchar eq {pop 27} if % bypass as comment if not used
dup 27 eq {exitproc} if scanstr 3 1 roll put } loop scanstr exch 0 exch getinterval gonzojustify
pop more not { exit} if} loop} bind def

/rslashproc {pop currentfile read not {pop /more false def exit} if
dup 41 eq { 1000 } if % replicate right paren
dup 40 eq { 1000 } if % replicate left paren
dup 92 eq { 1000 } if % replicate reverse slash
dup 98 eq { pop 8 1000 } if % substitute bs
dup 102 eq { pop 12 1000} if % substitute formfeed
dup 116 eq { pop 9 1000} if % substitute tab
dup 110 eq { pop /more true def exit} if % end string on linefeed
dup 114 eq { pop /more true def exit} if % end string on return
dup 48 ge { dup 55 le {dooctal} if} if
1000 eq { } {pop 1 sub dup 1 add 0 } ifelse % substitute or replicate} def

/dooctal {48 sub  64 mul /oct exch def currentfile read not {pop /more false def exit} if dup 48 ge
{ dup 55 le {dothird} {1000} ifelse} if} def

/dothird {48 sub 8 mul oct add /oct exch def currentfile read not {pop /more false def exit} if dup
48 ge { dup 55 le {48 sub oct add } if 1000} if} def

/exitproc { pop currentfile read not {pop /more false def exit} if dup exitchar eq {pop exit}{exch
scanstr exch 27 put exch 1 add dup 3 -1 roll} ifelse} def

% default font and variable list

% for drop caps
/font0 {/Helvetica-Bold findfont [40 0 0 40 0 0] makefont setfont} def

% regular text
/font1 {/Helvetica findfont [9 0 0 9 0 0] makefont setfont} def
/font2 {/Helvetica-BoldOblique findfont [9 0 0 9 0 0] makefont setfont} def
/font3 {/Helvetica-Bold findfont [9 0 0 9 0 0] makefont setfont} def

% slightly smaller for numbers and caps in text
/font4 {/Helvetica findfont [8 0 0 8 0 0] makefont setfont} def
/font5 {/Helvetica-BoldOblique findfont [8 0 0 8 0 0] makefont setfont} def
/font6 {/Helvetica-Bold findfont [8 0 0 8 0 0] makefont setfont} def

% lowered title line
/font7 {/Helvetica-Bold findfont [9 0 0 9 0 4] makefont setfont} def

% superscript and subscript
/font8 {/Helvetica-Bold findfont [6 0 0 6 0 0] makefont setfont} def
/font9 {/Helvetica-Bold findfont [6 0 0 6 0 4] makefont setfont} def

% specialty fonts
/font- {/ZapfDingbats findfont [9 0 0 9 0 4] makefont setfont} def
/font= {/Symbol findfont [9 0 0 9 0 4] makefont setfont} def

% default constants - redefine for each application

/justx (justL) def      % left justify
/txtwide 400 def        % width of column
/yinc 11 def            % line spacing
/charstretch  0.2 def   % minimum character kerning
/spacestretch 0 def     % minimum space kerning
/maxsstretch 2.5 def    % space fill stretch before character stretch
/maxcstretch 1 def      % maximum allowable fill character stretch
/dropcount 3 def        % lines indented for drop cap
/dropindent 40 def      % width reserved for drop cap
/pm 0 def               % paragraph margin
/pmnorm 10 def          % normal paragraph indent
/xpos 50 def            % horizontal start of text
/ypos 600 def           % vertical start of text
/colcheck { } def       % link to page or column maker

% here is a simple "just dump the text" pagebuilder. Preceed your text by -startgonzo- or [esc]-g
%  and end it with [esc]-s or [esc]-x -showpage- [d].

/ytop 600 def /ypos ytop def /ybot 50 def /colcheck { ypos ybot le {showpage /ypos ytop def} if}
def
```

Fig. 2C – Gonzo justify routines, concluded.

fusion machine, or even run it back through your LaserWriter, your toner image will *Bakerize*, turning into a glossy and dense black that is quite durable. Surprisingly so.

Try it for business cards – you'll be amazed at the results for a "zero cost" quick-and-dirty process.

What is Pixie Dust?

I have found one partial solution for the intolerably abysmal scratch resistance of the *Canon* type SX laser printer cartridges. The method seems to work a lot better than sticking pins into *Canon* advertisements.

The larger and older xerography machines will use a standard powder called *drum conditioner* that acts as a lubricant and noticably reduces, but does not outright eliminate all of the scratch problems.

Unfortunately, drum conditioner is normally available only in railroad tank car lots, and then only to a few approved insiders.

Arlin Shepard of *Laser Products* has repackaged the drum conditioner into a $3 packet of *Pixie Dust* that is enough for a dozen or more LaserWriter cartridges or their reloads.

Figure one shows you how to take a childs athletic sock and convert it into a pixie dust duster. What you do is *very lightly* dust the drum when you first install or later reload the cartridge. Be careful to not rotate or get any thumbprints on the drum when you do this.

After a few copies, most of the pixie dust will collect on the wiper and doctor blades and will continue to work from there.

Meanwhile, working on the theory that every little bit helps, do continue to stick the pins into all the *Canon* ads you see. Somebody, somehow just has got to the the attention of these epsilon minuses.

Tell me all About Your Gonzo Justify Routines

For several years now, I have been trying to push the limits of 300 DPI LaserWriter desktop publishing. We have seen in previous issues how everybody positively insists on using the seventeenth most putrid LaserWriter gray, and how that might be cured with a very few keystrokes. We have also found out how you can set legible type as small as three points through a "pixel locking" technique. We have already seen how *pixel line remapping* can paint practically any image onto most any surface.

But the centermost key to nearly everything, of course, lies in a decent justification routine that is able to attractively place mixed font column lines in order onto a page.

I have been continuously developing and debugging some *gonzo justify* routines that do what I want the way I want them to.

So, let me tell you what I think is important in a decent justify routine. Firstoff, I couldn't care less about WYSIWYG. At the present state of the art, all WYSIWYG does is slow you down and crap up all your final on-page images. This will, of course, dramatically change when everybody has and uses display PostScript.

I want a device independent gonzo justify routine that will run unmodified with *any* make and model of personal computer using *any* desired editor, word processor, or comm program. I want those same routines to be importable into nearly any PostScript applications program.

Above all, I do want to be able to import the full *AppleWriter* quality mixed font, fill justified typography into *Illustrator*.

I want a gonzo justify routine that is fully programmable. I want to be able to define or redefine any command and even the entire set at any time for any reason, instantly switching into or out of full PostScript. I also want a full macro capability that lets me use a single keystroke or a single embedded character to do such things as change from a centered, bold, lowered, non-indented title into a fill justified, a standard, and paragraph indented main text.

Naturally, I want to be able to use *any* mix of *any* size fonts on any particular line, including, of course, the ability to have fonts fatter than they are wide, and vice versa. I want fonts on, above, and below the baseline, and I want to lean them any amount in either direction.

Fractional font sizes are a must.

I want a four stage progressive fill microjustification. First, I want to add a definable minimum amount of kerning between each and every individual character. This is extremely important for 300 DPI printing in the 9 to 12 point range, yet nobody seems to offer it.

On stage two, I want to spread the spaces out some, only to a visually acceptable and attractive limit. On stage three, I want to further spread the individual characters out, again to a visually acceptable limit. Finally, if the first three stages will not hack it, I want to spread out the spaces as far

---- commands ----

cc
Callout center justify. Use as **xpos ypos (message) cc**
when not scanning. The centering is done on **xpos**,
and **txtwide** is ignored.

cf
Callout fill justify. Use as **xpos ypos (message) cf**
when not scanning. Uses **xpos** left margin, **txwide** width.

cl
Callout left justify. Use as **xpos ypos (message) cc**
when not scanning. Uses **xpos** left margin, **txwide** width.

cr
Callout right justify. Use as **xpos ypos (message) cc**
when not scanning. The right margin is set by **xpos**,
and **txwide** is ignored.

startgonzo
Begins gonzo scanning of text that follows. **[esc]-g**
may also be used. Terminates on end of file or **[esc]-x.**

[esc]-?
User defined embeddable escape commands and macro
commands. See figure four for the current list.

stringmacro
A shorthand way of building macros. For instance,
/amacro {(Cyz3) stringmacro} def does an **[esc]-C**
[esc]-y [esc]-z [esc]-3 all on a single **[esc]-a** embedded
command.

Fig. 3 – Some important gonzo commands and variables.

---- variables ----

ytop

as I have to. Later on, some post-justification editing can be used to return all the spacing back down to acceptable values.

I want the speed to not significantly slow down the LaserWriter NTX for simplier page layouts, and I want the ability to compile all the PostScript code back to the host for the fastest possible book-on-demand printing applications.

I want an automatic initial drop cap, plus the ability to use *hanging punctuation*. While rarely seen in anything but the finest of books and magazines, hanging punctuation can noticeably improve just about any 300 DPI fill justified text.

I want to be able to handle single "callout" messages as easily as I can large blocks of text. And I want the same routine to emulate just about anything that uses embedded printing commands. With the added ability to instantly switch between emulation and full PostScript at any time for any reason.

I want to be able to add bells and whistles such as tabs, French spaces, custom character kerning, extra end-of-paragraph ledding, soft hyphens, conditional formfeeds, hard spaces, superbold print, the works.

Finally, of course, I would want it fully open, totally unlocked, unprotected, very easily customized, and widely available at a very low cost.

So Where is it?

In figure two, of course. Figure two gives you a complete listing of all my current gonzo justify routines. Figure three lists all of the more important PostScript variables, while Figure four gives you a summary of the currently embeddable commands.

[esc] - 0	Change to **font0**. Preferred use = initial drop cap.
[esc] - 1	Change to **font1**. Preferred use = normal body text.
[esc] - 2	Change to **font2**. Preferred use = italic body text.
[esc] - 3	Change to **font3**. Preferred use = bold body text.
[esc] - 4	Change to **font4**. Preferred use = all caps normal body text.
[esc] - 5	Change to **font5**. Preferred use = all caps italic body text.
[esc] - 6	Change to **font6**. Preferred use = all caps bold body text.
[esc] - 7	Change to **font7**. Preferred use = bold and dropped headers.
[esc] - 8	Change to **font8**. Preferred use = subscripts.
[esc] - 9	Change to **font9**. Preferred use = superscripts.
[esc] - -	Change to **font-**. Preferred use = Zapf Dingbats.
[esc] - =	Change to **font=**. Preferred use = Symbol in body text size.
[esc] - C	Change to center justification on **xpos+txtwide/2**.
[esc] - F	Change to fill justification between **xpos** and **xpos+txtwide**.
[esc] - L	Change to left justification starting at **xpos**.
[esc] - R	Change to right justification ending at **xpos+txtwide**.
[esc] - a	Perform user defined Macro **amacro**.
[esc] - b	Perform user defined Macro **bmacro**.
[esc] - c	Perform user defined Macro **cmacro**.
[esc] - d	Perform user defined Macro **dmacro**.
[esc] - e	Perform user defined Macro **emacro**.
[esc] - f	Perform user defined Macro **fmacro**.
[esc] - g	Start gonzo justification. A PostScript alias for **startgonzo**.
[esc] - h	Do a half of a negative linefeed.
[esc] - i	Do an initial drop capital.
[esc] - l	Do a full linefeed.
[esc] - n	Nobreak - conditional formfeed if within six lines of **ybot**.
[esc] - p	Switch to normal paragraph margin, starting with next line.
[esc] - s	Do a **showpage** for end of scanned file only.
[esc] - x	Exit gonzo justify and switch to native PostScript.
[esc] - y	Do a **yinc** up full negative linefeed.
[esc] - z	Switch to zero paragraph margin, starting with next line.

Fig. 4 – Currently embeddable gonzo escape commands.

This *Ask The Guru* reprint is done using the gonzo justify. You might like to try using one of those heavy "name brand" layout applications packages to see if you can come up with results that are even remotely this good at 300 DPI.

We will be seeing lots more use examples in future columns. But for now, figure five gives you a simple column maker that produces your choice of 1:1 or oversize 1.33:1 columns. The latter can be photoreduced for "real" printing applications and will dramatically improve the overall smoothness of the characters.

Figure six gives you some of the text actually used in this column.

Since some more primitive word processors do not let you embed any escape commands, I have replaced each and every escape command with its **\033** octal equivalent.

This listing also does assume your word processor can handle up to a 240 character line without forcing extra carriage returns or linefeeds.

Additional reverse slash "ignore the following carriage return or linefeed" can be added if needed.

Note that carriage returns in figure seven are used only after an end-of-line reverse slash or following each paragraph. Extra white vertical space has been added in this figure to help you identify paragraph endings.

In general, the gonzo routines can either be tacked onto each individual application or else persistently downloaded when you first apply your power. Typically, you will need your gonzo justify, a template, and a text file. Your template will hold your page and column building routines as well as the font and style selections for your final copy.

A single **colcheck** command is used to link your gonzo justify routines back to your page- or column-making template.

For long blocks of text, there is a scanner built into the gonzo justify routines. This is activated through a **startgonzo** or an **[esc]-g** and can be exited at any point with an **[esc]-x**.

An ending **showpage** will often be needed after escaping, or you might miss the last page.

Macros are currently defined as **[esc]-a** through **[esc]-f**. These can do any PostScript function at all. There is also a supermacro command. For instance, the **/amacro {(z3C) stringmacro} def** command will switch you to a center justified, lowered and bold text with no paragraph indents, all on an **[esc]-a**.

The supermacro strings will work with anything already defined in the gonzolink dictionary. They may also be intermixed with the ordinary PostScript commands, but only gonzolink stuff is allowed inside the string.

If you just route plenty of text in without any template, a default page breaking text dumper will take over. This is handy for first proofs.

Alternately, you can now instantly position short messages by using a **xpos ypos (This is a test) cc** format. This will center your message onto your selected **x** and **y** positions. The options here include **cl**, **cf**, and **cr** for left, fill, and right justification.

The rule is to use plain text with the scanner and text delimited with parenthesis for callout commands.

With left or fill justification, the text is automatically broken up by words to fit between a left limit of **xpos** and a right margin limit set by **xpos+txtwide**.

There are two center justification and two right justification routines.

If you call for a center justify by using the **[esc]-C** command, then justification will take place between a left margin of **xpos** and a right margin set by **xpos+txtwide**, just like the usual left and fill justify. This is handy for centered headers.

If you call for a center justify by using **xpos ypos (string) cc**, then each line will end up *centered* on **xpos**, and **txtwide** will get ignored. This is useful for all of your ad copy headlines or anywhere else that you simply want to center on a position, independent of any column widths.

Similarly, an **[esc]-R** will print flush right up against **xpos+txtwide**, with no text ever to the left of **xpos**. And an **xpos ypos (string) cc** will simply align flush right against **xpos**. The right blurb on the *Ask The Guru* header does this.

How does it work?

There are four steps to gonzo text justification. These include *preparation*, *length analysis*, *justification calculations*, and finally *imaging*.

To prepare the text you will want justified, you isolate it as a string variable, and then present it to the **gonzojustify** routine, along with the position information, the choice of present font and your choice for the

```
% name of textfile: guru.temp.1 (Ask the Guru template)
% . . . . . . .

% requires gonzo.dl.2

/persist true def
/makeitbig false def

persist {serverdict begin 0 exitserver} if

/setscale {makeitbig{/scalefactor 4 3 div def 50 -300 translate scalefactor
dup scale  /ybotcorrect 200 def} {/scalefactor 1 def 0 0 translate
scalefactor dup scale /ybotcorrect 0 def } ifelse } def

/colcheck {ypos ybot ybotcorrect add le {showpage setscale header /ypos
ytop def /ystart ytop def } if} def

/header {/fontsave fontn 4 get def save /hdrsnap exch def /pm 0
def/yparendadj 0 def /xpos headerx def /xpos1 xpos def /ypos0 headery
def /ypos ypos0 def /txtwide headerwide def headertitle gonzojustify /ypos
ypos yinc add def pagenum 20 string cvs dup /num exch def
stringwidth pop /numwide exch def (page ) dup stringwidth pop
numwide add xpos txtwide add sub neg ypos moveto show num show
hdrsnap restore fontsave changefont /pagenum pagenum 1 add def} def

/font0 {/Times-Bold findfont [54 0 0 54 0 -32] makefont setfont} def
/font1 {/Times-Roman findfont [9.75 0 0 9.75 0 0] makefont setfont} def
/font2 {/Times-Italic findfont [9.75 0 0 9.75 0 0] makefont setfont} def
/font3 {/Times-Bold findfont [9.75 0 0 9.75 0 -6] makefont setfont} def
/font4 {/Times-Roman findfont [9 0 0 9 0 0] makefont setfont} def
/font5 {/Times-Italic findfont [9 0 0 9 0 0] makefont setfont} def
/font6 {/Times-Bold findfont [9 0 0 9 0 0] makefont setfont} def
/font7 {/Helvetica-Bold findfont [6 0 0 6 0 0] makefont setfont} def
/font8 {/Helvetica findfont [8 0 0 8 0 0] makefont setfont} def
/font9 {/Helvetica-Bold findfont [9 0 0 9 0 4] makefont setfont} def
/font- {/ZapfDingbats findfont [9 0 0 9 0 4] makefont setfont} def
/font= {/Symbol findfont [9 0 0 9 0 4] makefont setfont} def

/txtwide 155 def
/yinc 10.5 def
/charstretch  0.2 def
/spacestretch 0 def
/maxsstretch 2.5 def
/maxcstretch 1 def
/dropcount 3 def
/dropindent 18 def
/pm 0 def
/ytop 720 def
/ypos ytop def
/xpos 100 def
/headerx 80 def
/headery 760 def
/headerwide 240 def
/ybot 80 def
/yparendadj 0 def
/pmnorm 10 def

/amacro {(zy0)  stringmacro} def  % start drop cap
/bmacro {(iFy1) stringmacro} def  % finish drop cap
/cmacro {(zyC3) stringmacro} def  % centered title
/dmacro {(pF1)  stringmacro} def  % normal text
```

Fig. 5 – A gonzo column-builder "Guru" template.

present or previous justification style that you have requested.

The analysis always assumes you are going to want a fill justify. The input string is taken a character at a time, and all the individual character widths *for the presently selected font* are then added up as a running total.

```
% requires gonzo.dl.2 and guru.temp.1            NOTE: \033 = embedded [escape]
/makeitbig true def setscale  /headertitle (\0338Lancaster, Ask the Guru, Column 46) def
/pagenum 2 def /dropindent 36 def
% ///// ACTUAL TEXT STARTS HERE //////

header startgonzo

\033a
B
\033b

oy, a whole flock of them flew over that time. Apple has finally and officially flushed Pascal.
Pascal was basically a laboratory animal that escaped. Unfortunately, it did do an incredible \
and incalculable amount of damage to both academia and general computer literacy well before
its long overdue extermination.\033p

The heir apparent would seem to be the \0334"C"\0331 language, helped along with closely
linked and tightly writ- ten assembly language modules.

The \0334APW\0331 development system on the IIgs and the \0334MPW\0331 development
system on the Mac are good places to start using \0334"C"\0331. Both of these are \now
available through \0334APDA\0331.

What about \0334UNIX\0331 or \0334A/UX\0331? While \0334UNIX\0331 is written in
\0334"C"\0331, I personally feel that (a) It is insanely bloated, (b) It is already hopelessly \
obsolete, and (c) anything that \0334AT&T\0331 is in favor of has got to be suspect.

Let’s see. There is a new version \03343.2\0331 system master disk for the IIgs. More on this
shortly. The bottom line is that the printer driver improve- ments are far too little done far too \
late, but that there’s lots of other neat stuff included in the upgrade.

Quite a few of you helpline callers have been asking for ways to run PostScript on the low-end
dot matrix printers. Well, to do so would be the equivalent of putting a new \0332Porsche\0331
engine onto a skateboard.

Nonetheless, the \0332LaserGo\0331 people are the first ones out with such a product. Right
now, it is mainly for \0334IBM\0331 use, but Apple versions are cer- tain to follow.

\0332Apple Computer\0331 does have a few special interest publications that you might want
to be aware of. For you developers, there is the \0332Viewpoints\0331 newsletter and the \
\0332Outside Apple\0331 magazine.

For all of you bureaucrats, check into \0332Innovations\0331. For librarians, there is the
\0332Apple Library User’s Group\0331, and, among the many educational pub- lications, \
\0332Wheels for the Mind\0331 is just about the best.

Meanwhile, \0332Adobe Systems\0331 has a free \0332Font and Function\0331 typography
catalog out that includes useful design and layout ideas.

Turning to my own products, besides the \0332Ask the Guru\0331 reprints, we now also have
the \0332Hardware Hacker\0331 reprints from the sister column to this one over in
\0332Radio-Electronics\0331.

There’s a new disk for the gonzo justify stuff we’ll feature here, and, as usual, our
\0332PostScript Show and Tell\0331, available for most computers.

The gonzo biggie this month is just that \261 the gonzo justify I have been working on for
several years now. But first . . .

\033c
What is in the new IIgs
v3.2 System Master Disk?
\033d

Well, let’s see. First, it boots faster and is better organized, and literally hundreds of bugs have
newly been corrected. Or at least changed.

ProDOS \033416\0331 can now boot over AppleTalk, and the bugs in handling very large and
single byte files have been corrected. Random memory trashing has also been fixed, along \
with an easing of all the \0334SCSI\0331 format problems.

The printer drivers have been im- proved somewhat in a too little and too late manner. The
\0332ImageWriter\0331 driver is now faster on images and supports vertical condensed \
printing. A new and totally useless \0332ImageWriter LQ\0331 driver will recognize the machine
but not support any of its features.

While there are substantial new improvements in the LaserWriter drivers for applications willing
to play Apple games in Apple’s way, a plain old \0332Send PS\0331 is still conspicuously \
absent. You can, however, still rearrange \0334IWEM\0331 to make it into an acceptable
no-nonsense PostScript driver.\033x    showpage
```

Fig. 6 – Gonzo justify text example.

A minimum kerning that is set by **charstretch** and **spacestretch** is included in each width calculation.

The width addition continues until such time as (a) the next word will no longer fit on the line, or (b) you do change the font, or (c) you force a linefeed, or (d) you embed an escape command that does something fancy.

The end product of the analysis is an *array of substrings* that contains everything that will fit on the line.

Each substring is in a single font. The number of substrings needed per line depends on how fancy you get. If you make no font changes and embed no escape commands, then only a single substring is produced per line.

The justification calculations will depend on the selected justify mode. On a left justify, the substrings are simply printed as is. On a center or a right justify, the position is suitably shifted per the alignment you want.

On a fill justify, a four step calculation is made that decides exactly how much to stretch each space and each character to give you a true fill justify. Should hanging punctuation be needed, a "little extra" is added to put the punctuation out into the margin "just enough" that it looks good.

As a subtle touch, the last line in each paragraph is also stretched out slightly so it does not look cramped when compared against the previous lines. This stretching is defeated if you already are using more than 90 percent of the available linewidth.

Everything, of course, is fully programmable and changeable.

During imaging, a **compile** flag decides whether to print to paper or to return the minimum possible *PostScript* imaging commands back to the host for recording. Compiled code can be astonishingly fast.

We will see lots more on all this in future columns. Please note that the gonzo stuff is copyrighted code on which any and all commercial rights are fully and completely reserved.

You may copy it for your personal use provided the header remains intact and provided it is not included in any commercial software product or listing of any sort.

Yes, the ready-to-run gonzo justify disks are available in most formats for most personal computers. Write or call if you are interested.

Don Lancaster's

ASK THE GURU

November, 1989

Several hot stamping ideas
My computer workstations
Old and new game paddles
Using snoop and superexec
PostScript nuisance routine

No good deed shall ever go unpunished. At least in the computer publishing world. *Macintosh Today* recently went belly up because of a corporate bean counter who was upset with the rate of increase of ad revenue.

Worse still, Bob Sander-Cedarlof has ceased publishing his great *Apple Assembly Lines*, which was one of the finest Apple magazines anywhere ever. It seems that Bob finally let a few frivolous matters of food, clothing, and shelter get in his way. Bob does have a bunch of back issues and whatever available. Once gone, they are gone forever. You might contact him directly for more information.

Steve Ciarcia is leaving *BYTE* to go whole hog on his own already great *Circuit Cellar Ink* magazine. His final November and December *BYTE* columns are absolute mind-blowers that involve a parallel processor that produces *Mandlebrot* fractal images in *near real time!*

Which means it passes a *Cray-1* like it was sitting up on blocks.

Apple has finally addressed the disk drive blowup problems on the IIgs. If you ever do experience any difficulties with both drives coming on at once or have similar hangups, see your dealer for a free change of your disk drive internal I/O board.

Let's see. *Apple* now has two new videos out, a user group one on Woz talking about me, and one on desktop publishing and communication solutions. There is also a new *Desktop Solutions* portfolio that goes with the second video. See your user group for more details.

Two publications included here are the fat *Apple Desktop Communications Solutions Reference Guide* and their *DeskTopics* newsletter. Be sure to check these out.

As usual, you can get a list of all of your local user groups by calling (800) 538-9696, Extension 500.

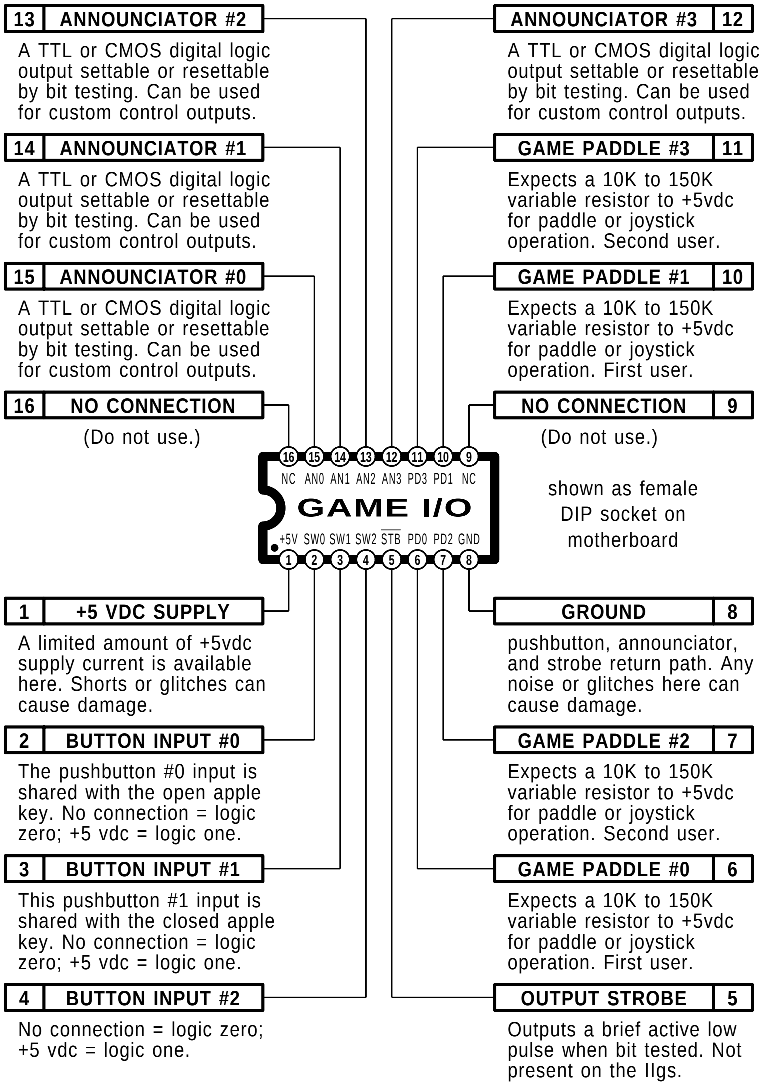
The PostScript bulletin board that was previously at (409) 244-4704 is going through some reorganization difficulties. It should shortly reopen in a major city with a new number, a larger hard disk, multiple lines, and both *Telenet* and *Tymenet* access.

While you are waiting, you might want to check out a brand new PostScript board at (707) 882-2390.

U.S. Patent #4740443 shows you how to implant carbide teeth onto your toner particles so you can now scratch and grind up all of your laser printer cartridge drums. Guess which company it was issued to?

For this month's lukewarm stock tip, *Seagate Technology* had recently overexpanded a tad and the institutions have all pressed their collective panic buttons. An instutitional investor would now rather carry AIDS than SGAT. Buy at $8; sell at $18.

There have been a dozen minor corrections to the gonzo justify we looked at last month. I've also now added custom kerning, tabs, and an improved compile to the routines.



13 ANNOUNCIATOR #2
A TTL or CMOS digital logic output settable or resettable by bit testing. Can be used for custom control outputs.
14 ANNOUNCIATOR #1
A TTL or CMOS digital logic output settable or resettable by bit testing. Can be used for custom control outputs.
15 ANNOUNCIATOR #0
A TTL or CMOS digital logic output settable or resettable by bit testing. Can be used for custom control outputs.
16 NO CONNECTION
(Do not use.)
1 +5 VDC SUPPLY
A limited amount of +5vdc supply current is available here. Shorts or glitches can cause damage.
2 BUTTON INPUT #0
The pushbutton #0 input is shared with the open apple key. No connection = logic zero; +5 vdc = logic one.
3 BUTTON INPUT #1
This pushbutton #1 input is shared with the closed apple key. No connection = logic zero; +5 vdc = logic one.
4 BUTTON INPUT #2
No connection = logic zero; +5 vdc = logic one.
ANNOUNCIATOR #3 12
A TTL or CMOS digital logic output settable or resettable by bit testing. Can be used for custom control outputs.
GAME PADDLE #3 11
Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. Second user.
GAME PADDLE #1 10
Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. First user.
NO CONNECTION 9
(Do not use.)
16 15 14 13 12 11 10 9
NC AN0 AN1 AN2 AN3 PD3 PD1 NC
GAME I/O
+5V SW0 SW1 SW2 $\overline{\text{STB}}$ PD0 PD2 GND
1 2 3 4 5 6 7 8
shown as female DIP socket on motherboard
GROUND 8
pushbutton, announciator, and strobe return path. Any noise or glitches here can cause damage.
GAME PADDLE #2 7
Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. Second user.
GAME PADDLE #0 6
Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. First user.
OUTPUT STROBE 5
Outputs a brief active low pulse when bit tested. Not present on the IIgs.

Fig. 1 – Connections to the "old" Apple Game Socket.

Volume II of *Ask The Guru* is now shipping. This is a complete rewrite including lots of new material, a big master *Names and Numbers* section, and a full index. Besides all of the Apple and PostScript stuff, the text is also a tour-de-force of personal book self-publishing.

As per usual, this is your column and you can get tech help and some off-the-wall networking per the *Need Help?* end box. And now . . .

What Computers Do You Actually Use?

About a dozen of them, I'd guess. But my main machine is now a cross between a IIe and a IIgs running a fast ProDOS 8 and driving a *LaserWriter NTX*. *AppleWriter* and *WPL* are run on this machine virtually all of the time.

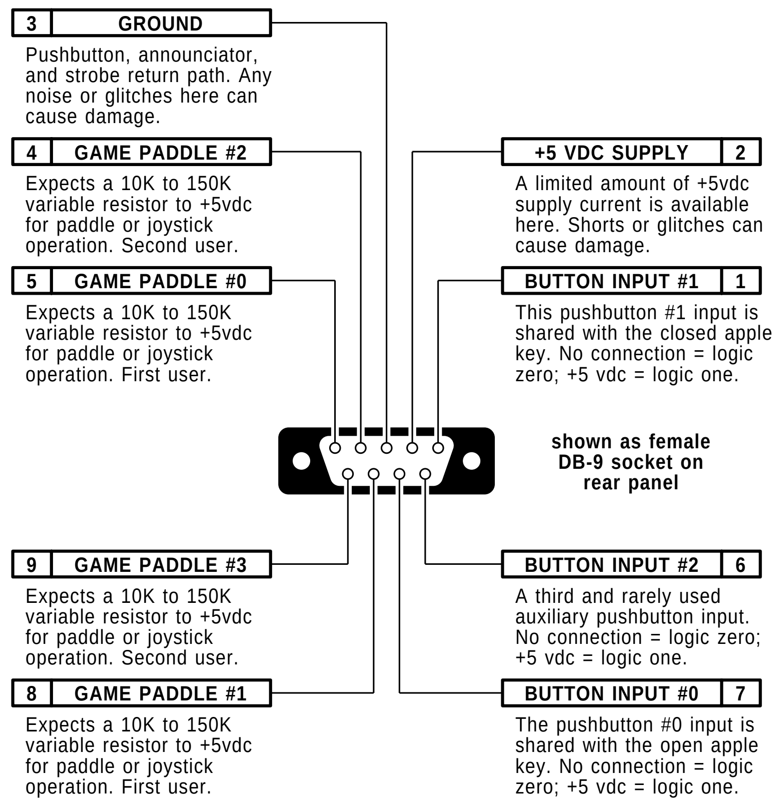
Bee has a IIc (she actually *likes* that insidious keyboard) driving a *LaserWriter Plus* with a modem for her personal workstation. Software here includes *AppleWriter*, production copy utilities, accounting stuff, and some weaving goodies.

Our book-on-demand production setup is a IIgs once again driving a *LaserWriter Plus*. Sadly, that 15:1 per-page toner cost penalty on the NTX prohibits us from using it here. Instead, I am still stuck with one of my ancient 400,000 copy *LaserWriter Plus* machines.

At first glance, some of this may seem a tad on the wimpy side. But note that I can run portions of my PostScript book production routines as much as *one hundred times* faster than can a novice user running their canned application programs on a *Macintosh II.*

For instance, a very fancy three column gonzo-justified page having two figures, a header, and a footer, needs a mere *zero to four seconds* of extra page composition time above the NTX "full tilt" printing speed.

I very much prefer a computer in which I can understand and freely modify each and every level of hardware and software in the machine to suit myself.



3 GROUND: Pushbutton, announciator, and strobe return path. Any noise or glitches here can cause damage.

4 GAME PADDLE #2: Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. Second user.

5 GAME PADDLE #0: Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. First user.

+5 VDC SUPPLY 2: A limited amount of +5vdc supply current is available here. Shorts or glitches can cause damage.

BUTTON INPUT #1 1: This pushbutton #1 input is shared with the closed apple key. No connection = logic zero; +5 vdc = logic one.

9 GAME PADDLE #3: Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. Second user.

8 GAME PADDLE #1: Expects a 10K to 150K variable resistor to +5vdc for paddle or joystick operation. First user.

BUTTON INPUT #2 6: A third and rarely used auxiliary pushbutton input. No connection = logic zero; +5 vdc = logic one.

BUTTON INPUT #0 7: The pushbutton #0 input is shared with the open apple key. No connection = logic zero; +5 vdc = logic one.

Fig. 2 – Connections to the "new" Apple Game Connector.

Yes we have a Mac. It is used once a month to handle our PostScript disk production and translations. Otherwise, it serves as a handy doorstop, and Orville the cat sure does like to sleep on it.

Kathy, who runs a service bureau down the street, uses a IIgs and a II+ to do our order processing, shipping label printing, and all our mailing list maintenence. Clone translations and production are done across town, and the *Atari* stuff is handled by a Tucson caver via modem.

I also still do use six of the *KIM-1* computers, one of which does have a full video and EPROM burning capability. To this day, there is no finer computer to teach the fundamentals of machine language programming. I personally feel that the KIM-1 was the last decent computer that *Commodore* ever built, and sure would like to see it reissued.

Finally, since my personal computer collection is so skimpy, I am building up some of my own. These do include a little battery powered $50 control computer using that great *Mitsibushi* M50734. The M50734 is a latter-day 6502 that does include such built-in goodies as 40 I/O lines, 257 accumulators, four A/D converters, two stepper motor drivers, a watchdog circuit, four timers, a UART, a pulse position modulator, and a soft ice cream dispenser.

What's the Difference Between the old and new IIe and IIgs Game Paddle Connectors?

The original Apple II and II+ had a game paddle connector that consisted of an ordinary 16 pin DIP socket on the motherboard. Which had the dual disadvantages of the game paddle or joystick pins that were really easy to break, and that you had to remove the lid of your Apple every time that you wanted to plug or unplug your paddles or joystick.

The full sixteen pins did give you four game paddles or two joystick inputs, three pushbutton inputs, four announciator outputs and a strobe output signal. A secret fourth pushbutton input was available if you did look around for it hard enough. Hint: override the casssette input at the output of its op-amp.

Figure one shows you this "old"

game socket and its connections.

With the IIc, a "new" 9 pin DB-9 back panel connector was introduced. This let you use much more rugged connectors on your paddles and joysticks, and now did let you plug or unplug without taking the machine apart. On the other hand, it dropped the four annunciator outputs and the strobe, and all of those really neat "quick and dirty" I/O tricks you could do with them.

Figure two shows you this "new" game socket and its connections.

The Apple IIe did give you both connectors and let you have the best of both possible worlds. In theory, so did the IIgs. Only early versions of the IIgs left off the strobe output for some inexplicable reason. Strobe just plain is not there. Hopefully, this will someday be restored.

At any rate, you might like to use old paddles on a new machine or vice versa. Figure three shows you the two adapters you will need to either get from the old paddles to a new connector, or from new paddles to an old connector.

Since the "old" IIgs game paddle connector is far and away the fastest way of doing your 57600 baud serial *LaserWriter* communications (this is insanely faster than is use of Apple-Talk), there seems to be a bunch of renewed interest in what you can *really* do with that internal game connector. More on this whenever.

One tip: If you are still using the "old" paddles or joysticks, add a 16 pin *machined pin contact* DIP socket to the pins. This can then safely be plugged into the motherboard. Now, when (not if) the pins bend or break off, all you have to do is snap on another DIP socket, rather than taking the paddle or joystick apart and trying to fix it.

Another tip: If you are reading a pair of paddles or a joystick from machine language, be sure to wait several milliseconds between your successive paddle or joystick reads; otherwise, you can get some strange interaction between the two paddles or the joystick "X" and "Y" axes.

Tell Me about Hot Stamping

Our never ending quest for wierder and more bizarrely wonderous "neat stuff" goodies for laser printers has finally led us around to hot stamping.

While you can use *Omnicolor* or *Kroy Kolor* to fake a hot stamping, the real thing is far more durable and far more impressive.

Hot stamped foil consists of a thin and brightly colored film or foil to which a thin layer of high temperature adhesive has been attached. You press a stick of type or else a steel or silicon rubber die against the foil and quickly heat it. The heat and pressure then transfers the foil to the paper or other substrate. The cost is around a dime a square foot.

Obvious uses do include business cards, book covers, announcements, ad displays, and greeting cards. Or anywhere else you want to impart a high quality image.

Hot stamping is sometimes combined with embossing or debossing, often with spectacular results.

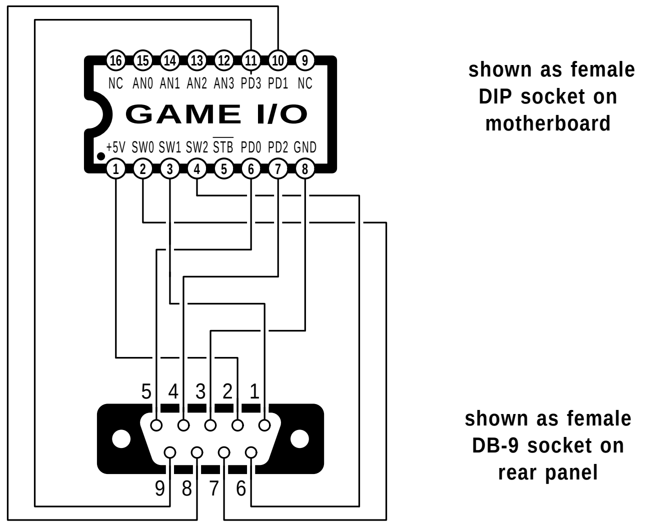
Here's what I have found out so far: Sources of foil and hot stamping machines advertise regularly in both *Paper, Film, and Foil Converter* and in *Package Printing and Converting* magazines. A house newsletter with one incredible title is *Foiled Again* and is published by *Transfer Print Foils*; these people also supply a new product called *All Purpose Roll Leaf.*

Some of their competitors include *Maple Roll Leaf*, *Coburn*, *Lamart*, *Identicolor* and *Hoechst/Hostaphan*. Finally, those *Bind-It* people have a desktop *Foil-It* system that does seem obscenely overpriced at $800, but is otherwise interesting.

We saw in a previous column how that *Merigraph* ultraviolet curing photopolymer process will let you convert a laser printed image into a rubber stamp or other printing die.

The chances are you would have to make a silicon rubber die off the Merigraph die in order to get proper high temperature operation. But the potential opportunities here boggle the mind. Neat stuff.

I have this strange feeling that I am only scratching the surface somewhere way out in left field. So, why don't you tell me about sources and supplies of foil stamping, or of the many brand new things that you can do with them that does involve laser printing? We'll have another one of our stupid contests. An *Incredible Secret Money Machine* for the best twenty entries, and an all expense paid (FOB Thatcher, AZ) *tinaja quest* for two for the best response of all.


16 15 14 13 12 11 10 9
NC AN0 AN1 AN2 AN3 PD3 PD1 NC
GAME I/O
+5V SW0 SW1 SW2 STB PD0 PD2 GND
1 2 3 4 5 6 7 8
shown as female DIP socket on motherboard
5 4 3 2 1
9 8 7 6
shown as female DB-9 socket on rear panel

Use a **female** 24 pin DIP socket and a **male** DB-9 plug to adapt an **old** game paddle or joystick to the rear panel connector.

Use a **male** 24 pin DIP plug and a **female** DB-9 plug to adapt a **new** game paddle or joystick to the internal motherboard socket.

Fig. 3 – Two old/new game paddle adapters.

What is This Month's PostScript Utility?

Two of the real beauties of the PostScript language are that it is both *threaded* and *extensible*. This means you can add any new commands you want to the language any time you wish. The only rule is that you have to define any of your new commands by using combinations of previously defined ones.

acos	Trig arccosine. call with **xside hypotenuse acos**. See also **asin**.
asin	Trig arcsine. call with **yside hypotenuse acos**. See also **acos**.
backwards	Print backwards from right to left. For transparent window decals.
baud.57600.x.25	Set the baud rate to 57600 baud, XON/XOFF handshaking, on the DB25 connector. The others are similar. This is device dependent.
bestgray	A 106 line, 45 degree screen. A good general, all purpose fine gray.
black	Switch to black. Same as **0 setgray**.
blackflash	Eject totally black page. Use twice for highest possible print quality.
copies	Select the number of copies, as in **6 copies**.
dobaud	A service routine used by **baud.57600.x.25** and company.
feetfirst	Eject the page rightside up or feet first.
inch	Change size to inches, as in **3 inch** for a 3 inch line.
indiagray	A very fine 133 line, 25 degree gray with 6 gray levels. Requires care.
landscape	Switch to landscape mode, printing wider than high.
lightgray	Select the lightest available gray.
listfonts	Send a list of all resident fonts back to the host.
longjob	Lengthen the wait timeout to 3 minutes. Gives the host extra disk time.
manual	Select manual, rather than tray feed. For thick or oddball paper stock.
negative	Print white as black and black as white. Useful for reversals.
notestpage	Cancel the test page that is printed on startup.
outline	Find the outline path of the current character.
persist	Do a persistent download of the code that follows.
pi	The math constant 3.1415926.
pixel	Scale to 300 DPI pixels, rather than points.
positive	Restore normal printing after using **negative**.
printfonts	Print a list of all resident fonts onto the page.
putridgray	Restore the default 53 DPI, 45 degree gray screen.
random	Generate a random number. **6 random** returns 0-5.
report	Return top of stack to host and stall briefly.
reprogray	A good 85 DPI, 35 degree gray for reduced prepress output.
resetttimer	Initialize multiple time interval measurement. See **starttimer**, **stoptimer**, and **reporttimer**. Use **stopwatchon** and **stopwatchoff** for single shots. For instance, two time intervals would be measured by **resettimer**, **starttimer**, {do something} **stoptimer**, {spin wheels} **starttimer** {do something} **stoptimer reporttimer**.
reporttimer	Return measured time to host. See **resettimer**.
snoop	Activate **superexec**. A **{forall} superexec** overrides **readonly**.
stall	Time delay, as in **1500 stall**. Gives host time to respond or record.
starttimer	Begin part of multiple time measurement. See **resettimer**.
stoptimer	End part of multiple time measurement. See **resettimer**.
stopwatchoff	End single time measurement and report to host. See **resettimer**.
stopwatchon	Reset and begin single time measurement. See **resettimer**.
tan	Trig tangent. Use **yside xside tan**. Inaccurate very close to 90 degrees.
tray	Return to normal tray feed.
white	Switch to white. Same as **1 setgray**.
width	A **(string) width** finds only the x width of the string.
yestestpage	Restore the test page that is printed on startup.

Fig. 4 – Some new PostScript nuisance commands.

So, what I have come up with is a new *nuisance dictionary* that makes doing great heaping bunches of tricky things (such as turning off that test page) easily done with a single command. What you do is persistently download this dictionary or else tack it onto anything else you are persistently downloading on power up.

From that point forward, you can activate all of these new commands with a simple **nuisance begin**.

Figure four shows you a detailed list and explanation of what all the nuisance commands do, while figure five will give you all of the working code ready to download.

Included are new trig commands for the tangent, arccosin and arcsin; one-word choices for backwards, feet first, frontwards, and landscape printing; excellent gray selections; easier baud rate and handshaking set commands; simpler choice of the number of copies and job length; a persistent downloading; instant listing of currently resident fonts to host or page; a sublime sneaky secret snooper that activates **superexec**; a few repeatable timers and stopwatches; a quite easy random number generator; and a one word stopping of the test page.

Some of these new commands are specific to the *LaserWriter*, while a few others are intended primarily for 300 DPI printing. All the rest can be used pretty near anywhere.

The *blackflash* command ejects a totally black page. For the highest possible print quality, you eject two fully black pages before you do print your final copy. This gives you some excellent black fills and light grays that are all highly uniform.

Speaking of which, there are now four commands to pick useful grays, instead of the seventeenth lousiest one that most PostScript applications seem to have their heart set on. The **bestgray** is the all around best choice here; a denser **indiagray** is available but requires care in use, along with a **reprogray** that is ideal for reduced prepress. If you ever want to return to the default gray, just use the new **putridgray** command.

There are now six different timer commands, depending on what you want to do. If you are only interested in making one single time measurement, you can use **stopwatchon** and **stopwatchoff**. If instead, you wantto find out the accumulated time that gets spent inside certain PostScript procs, you start off with a **resettimer**. Then you do a **starttimer** every time you enter the proc, followed by a **stoptimer** on each exit. Finally, you report the time back to the host with a **reporttimer** command.

The **snoop** command will activate **superexec** for your use. For instance, if a **forall** gives you an **invalidaccess** error, a **{forall} superexec** often will not. The same trick will often work for **{get} superexec**.

Calling **snoop** also does open up **internaldict** along with all its many strange and wondrous denizens. Very handy. And most interesting.

One of the extremely interesting routines in *internaldict* is *FlxProc*. This takes a pair of cubic splines and using a highly unique erosion algorithm, converts any curved lines to straight or "less bent" ones as their size diminishes below a crucial value. This is one key to PostScript's ability to attractively show any very small typography. And *snoop* lets you view it anytime you like.

A **6 random** command will return to you a random number in the range of zero to five. Should you in fact really want a one to six result, just tack on a **1 add** after calling **random**.

Two listers of resident fonts are provided. The **listfonts** command will return a list of all the currently resident PostScript fonts (these may be internal, custom created, or downloaded) back to the host. The **printfonts** will in- stead print a list directly onto paper for you.

The testpage can be defeated with **notestpage** and later restarted by activating my **yestestpage** command.

These several dozen brand new commands are my choice of what I presently think I would like for me. More of these goodies will almost certainly be along later. Obviously, you can easily and permanently rearrange the scenery to suit yourself.

So, why don't you? As yet another stupid contest this month, just show me how to improve upon any of the existing nuisance commands or add some new ones that others might find useful. Let's hear from you.

```
%  Copyright c 1988 by Don Lancaster and Synergetics, Box 809, Thatcher, AZ 85552.
%  (602) 428-4073. All rights reserved. Personal, non-commercial use permitted so long as
%  this header remains present and intact. Latest disk costs $24.50.

%  nuisancedict is persistently downloaded as a dictionary. Activate with nuisance begin

0 exitserver serverdict begin 200 dict /nuisance exch def nuisance begin. 1000 dict /nuisance
exch def nuisance begin

/acos {2 copy dup mul exch dup mul sub sqrt exch pop exch atan} def % xside hypot acos

/asin {2 copy dup mul exch dup mul sub sqrt exch pop atan} def % yside hypot asin

/backwards { 612 0 translate  -1 1 scale} % print backwards

/baud.1200.d.25 {1200 25 4 dobaud} def /baud.1200.x.25 {1200 25 0 dobaud} def
/baud.9600.d.25 {9600 25 4 dobaud} def /baud.9600.x.25 {9600 25 0 dobaud} def
/baud.19200.d.25 {19200 25 4 dobaud} def /baud.19200.x.25 {19200 25 0 dobaud} def
/baud.38400.d.25 {38400 25 4 dobaud} def /baud.38400.x.25 {38400 25 0 dobaud} def
/baud.57600.d.25 {57600 25 4 dobaud} def /baud.57600.x.25 {57600 25 0 dobaud} def

/bestgray {106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen} def % 106 line

/black {0 setgray} def  /lightgray {0.99 setgray} def  /white {1 setgray} def

/blackflash {0 0 moveto 1000 0 rlineto 0 1000 rlineto -1000 0 rlineto closepath fill showpage }
def % black pre-page for highest print quality

/copies { /#copies exch def} def  % as in 6 copies

/dobaud {serverdict begin 0 exitserver statusdict begin 3 copy setsccbatch setsccinteractive
end quit} def  % used by baud commands

/feetfirst {180 rotate 612 792 translate} def % eject print feet first

/inch {72 mul} def  /pixel {72 mul 300 div} def % 300 dpi only

/indiagray {135 35 {dup mul exch dup mul add 1.0 exch sub} setscreen} def %133 screen

/landscape {-90 rotate -792 0 translate} def  % pick landscape printing

/listfonts {FontDirectory {pop == flush 200 {37 sin pop} repeat } forall} def % fonts to host

/longjob {statusdict waittimeout 180 put} def % lengthen job timeout

/manual {statusdict /manualfeed true put} def % start manual feed

/negative {{1 sub abs} settransfer} def  /positive {{ } settransfer} def

/notestpage {serverdict begin 0 exitserver statusdict begin false setdostartpage end end quit}
def % cancel test page

/outline {false charpath} def % finds character outline path

/persist {serverdict begin 0 exitserver} def % starts persistent download

/pi 3.1415926 def % you wanted rhubarb instead?

/printfonts {/Helvetica findfont [10 0 0 10 0 0] makefont setfont /xpos 150 def /ypos 600 def
/yinc 12 def xpos 20 sub ypos 20 add moveto (CURRENTLY INSTALLED FONTS:) show
FontDirectory {pop 100 string cvs xpos ypos moveto (/) show show /ypos ypos 12 sub def}
forall showpage} def % fonts to page

/putridgray {53 45 {dup mul exch dup mul add 1.0 exch sub} setscreen} def % stock screen

/random {rand 65536 div 32768 div mul cvi} def  % as in 6 random for 0-5 result

/report {== flush 100 {37 sin pop} repeat } def % top of stack to host

/reprogray {85 35 {dup mul exch dup mul add 1.0 exch sub} setscreen} def % 85 line

/reportimer {mytime 1000 div (\rElapsed time: ) print 20 string cvs print ( seconds.\r) print
flush} def % to host

/resettimer {/mytime 0 def} def % reset timer

/snoop {1183615869 internaldict begin} def  % activates superexec

/stall {{37 sin pop} repeat} def % delay as in 1500 stall

/starttimer {usertime /mytimenow exch def}  % add to time so far

/stoptimer {usertime mytimenow sub /mytime exch mytime add def} def % for multiple timing

/stopwatchoff {stoptimer reporttimer} def  /stopwatchon {resettimer starttimer} def

/tan {dup sin exch cos dup 0 eq {pop 0.000001} if div} def % tangent

/tray  {statusdict /manualfeed false put} def % stop manual feed

/width {stringwidth pop} def % finds x width of string

/yestestpage {serverdict begin 0 exitserver statusdict begin true setdostartpage end end quit}
def % restore test page

end
```

Fig. 5 – A PostScript nuisance dictionary.

Don Lancaster's

ASK THE GURU

December, 1989

Apple game paddle circuits
Book publishing on demand
PostScript intelligent directory
The GOCCO silk screen process
LaserJet to LaserWriter conversion

Apple has just introduced a new version of the IIc. It includes an internal power supply and built-in 3.5 inch drive and runs up to 3.3 times faster. The price was also lowered slightly.

The memory expension connector has been changed. Third-party memory expansions to one megabyte and beyond by *Applied Engineering* and the "usual suspects" are expected.

They also further upgraded the IIgs operating system to version 4.0. This is the first time a true and full 16-Bit IIgs operating system was provided. Outside of it running considerably faster and fixing several bugs, there are not too many changes over the recently released version 3.2 system master. An upgrade kit is available from your dealer for $39. Plans are afoot to make the manuals and the documentation available through the usual BBS and user group sources.

Prices of most memory-intensive Apple products were also sharply raised, now that the RAM chip prices are dropping dramatically and the RAM shortage is ending. They also introduced a pricey Mac IIx with – wonder of wonders – firmware in a SIMM plug in module. This miraculously can let you flush QuickDraw and replace it with a third party Display PostScript. Whoopee!

There are two new Apple related publications out. One is *REBOOT*. It certainly gets the record for the northernmost Apple newsletter, since any further north from Unalakleet would be south. This is very readable, particularly for novice users.

II Technical is a new magazine for all of you intermediate to advanced Appleholics. Free samples are now available. Contact Brian Fitzgerald or Greg Autry for your personal copy.

Adobe Systems is having a big sale on their high quality downloadable PostScript fonts. Buy a font for only $32 and get a 45 Megabyte hard disk thrown in free. Its called the *Adobe Type Folio*, and the only tiny gotcha is that you have to buy all 300 fonts at the same time. It still is a real bargain for high end NTX owners that want a complete set of the finest available typography.

Adobe also has a free *Font and Function* printed listing and a free *Adobe Type Stack* Mac Hypercard disk available.

Did you hear the one about IBM making a minor change to one page of their DOS manual and then charging users $125 for the upgrade?

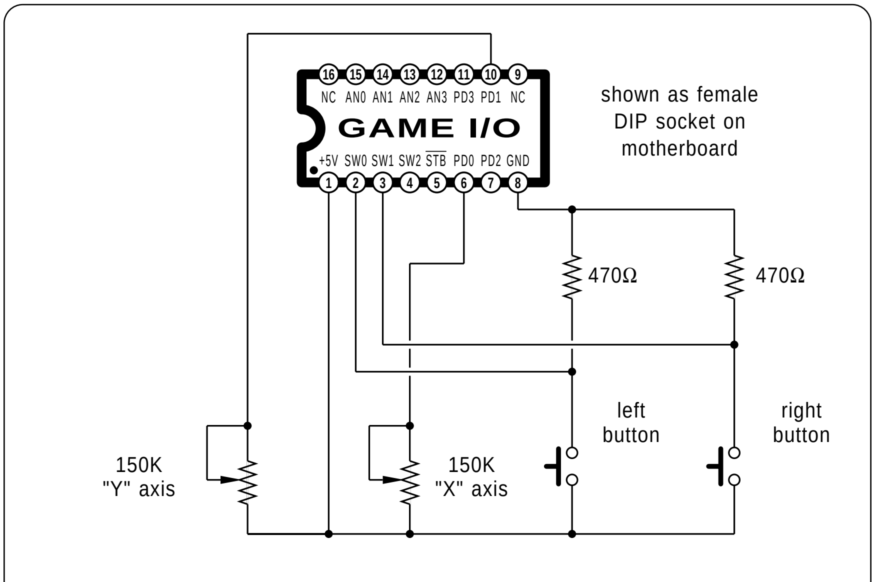


Fig. 1 – Circuit for the "old" Apple Game Paddles or Joystick.

Or that the FCC is demanding that all future high definition video displays in this country be strictly and absolutely compatible with the obsolete, 30 year old NTSC (Never The Same Color) broadcast video?

This is exactly the same as asking a consortium of trolley car manufacturers to dictate manditory rules and regulations for all personal vehicles of the nineties. Run a stick over the cage bars of these epsilon minuses if you are as appalled over this blatant atrocity as I am.

Let's see. We got *Ask the Guru*, volume I and II in print now, along with a *Hardware Hacker* volume II, both published on demand. Stuff you just plain will not find elsewhere. Along with all the usual PostScript stuff. Write or call for info.

As per usual, this is your column and you can get tech help and off-the-wall networking per the number in the end box. Also per usual, all of the names and numbers are gathered together at the end of the column.

Back to some basics . . .

How do the Apple Game Paddles Work?

Figure one shows you both the schematic and connections used for the "old" 16-pin Apple game paddles, while figure two shows you the same for the "new" 9-pin paddles.

Joysticks are similar, with the exception that a pair of potentiometers are mechanically linked together into one assembly.

The pushbuttons work by bringing their respective button lines high to +5 volts when they are activated. Note that PB0 gets shared with the *open-apple* key and PB1 gets shared with the *closed-apple* key on all the newer machines. Note also that a defective game paddle, joystick, or graphics tablet can simulate a permanant holding down of one of the apple keys, doing some wildly wrong things to *AppleWorks*, to *AppleWriter* or to several other programs.

Pushbutton #0 or the open-apple key is read by doing a PEEK (-16287) from Basic or else a BIT $C061 from machine language. A result greater than 127 or a set N flag means the button or key is currently pressed.

Similarly, either Pushbutton #1 or the closed-apple key is read by doing a PEEK (-16286) from Basic or a BIT $C062 from machine language.

The paddle or joystick potentiometers must be 150K units *that will actually change from 0 to 150K over their mechanically active range*. Note that other resistance values and other standards are used on other personal computers; as a general rule, the paddles or joystick must be matched to the computer they are run on.

The potentiometers form the variable charging resistor in a quad timer circuit that is in the 555 chip family. At the start of a timing interval, a timing capacitor is allowed to start charging. The timing interval ends when the capacitor reaches a preset value. By measuring the charging time, you can relate the paddle or joystick position to a number in the range of 0-255.

Automatic paddle measurement is done from Basic with an X = PDL (0) or Y = PDL (1). From machine language, a JSR $FB1E will read the paddle number whose value is in the *X* register at the time of the subroutine call. An $00 to $FF value is returned in the accumulator.

One gotcha: if you do try to read successive paddles too close together in machine language you will get some interaction. Always be sure to wait several milliseconds before any successive paddle reads. Applesloth Basic is so slow that it automatically does this for you.

Yes, you can modify your game paddles. The II+, IIe, and IIgs give you a third and fourth paddle input and a third pushbutton input. You can also fake a fourth pushbutton input with an extra wire on the II+ or IIe by jumpering to the output of the cassette read comparator.

To write your own routines for, say, an A/D converter or for extra resolution, you can use BIT $C070 to reset the paddle circuitry and a BIT $C064 to sense paddle zero, or else a BIT $CO65 to sense paddle one.

It is also possible to use the paddle inputs as plain old logic or pushbutton inputs that are compatible with any model Apple. This is a tad on the slow side since you have a high capacitance input and need a delay to verify your input, but it certainly will work.

Can I convert a LaserJet Into a LaserWriter?

P.T. Barnum was right.

The only two little fatal flaws in the *Hewlett-Packard* LaserJet series of printers are that they are totally brain dead and, as factory stock, are incapable of speaking PostScript. In my opinion, these fatal flaws make all of the LaserJets totally useless for any serious desktop publishing.

Many LaserJet users apparently agree, for they are finally realizing how bad they have been had and are flooding our helpline with pleas for help salvaging their investment.

Let’s start off with some positive comments: The LaserJets and the LaserWriters both use nearly identical and very well performing *Canon* CX or the newer SX printing engines. Sadly, these engines represent only around 15 percent of a laser printer; it is the PostScript *raster image processor* that does all of the important work and ends up getting nearly all of the useful results.

The slide-in toner cartridges are fully interchangeable between similar series of LaserJets and LaserWriters. But note that *you can not and must not attempt swapping a laser printer cartridge with a personal copier one.* The toner chemistry, light sensitivity, and interlocks are totally different.

Yes the cartridges, particularly on those older CX machines, are easily refillable several times for as little as $7.50 per refill, as we have seen in past columns and in both volumes of *Ask The Guru*.

HP’s attitude towards service and repair parts are light years ahead of the black hole policy at Apple, and the more knowledgeable LaserWriter users end up using LaserJet manuals and HP repair parts, available via VISA and an 800 number overnight to anyone anytime. The applicable older CX manual goes by HP part number #02686-90904, while their newer SX manual is #33440-90904.

There are half a dozen reasonable routes to salvaging your LaserJet investment that would seem more productive than staking your nearest HP salesman to an anthill.

One is to snap on an intelligent and genuine PostScript speaking lid. The *QMS* people offer several *PS-Jet* lids that give you full PostScript capabilities. Unfortunately, these lids can cost considerably more than your LaserJet, and the combined price can end up rather on the high side.

Weitek now provides some super

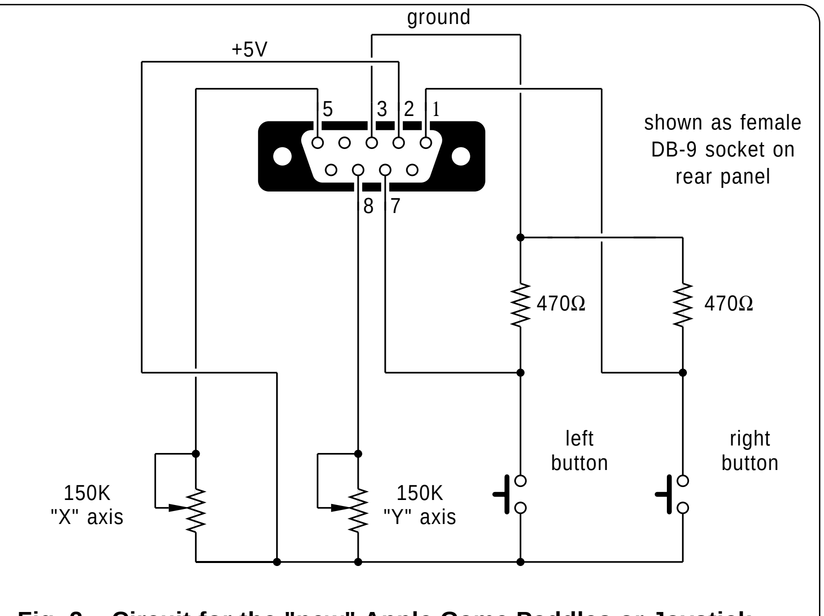


Fig. 2 – Circuit for the "new" Apple Game Paddles or Joystick.

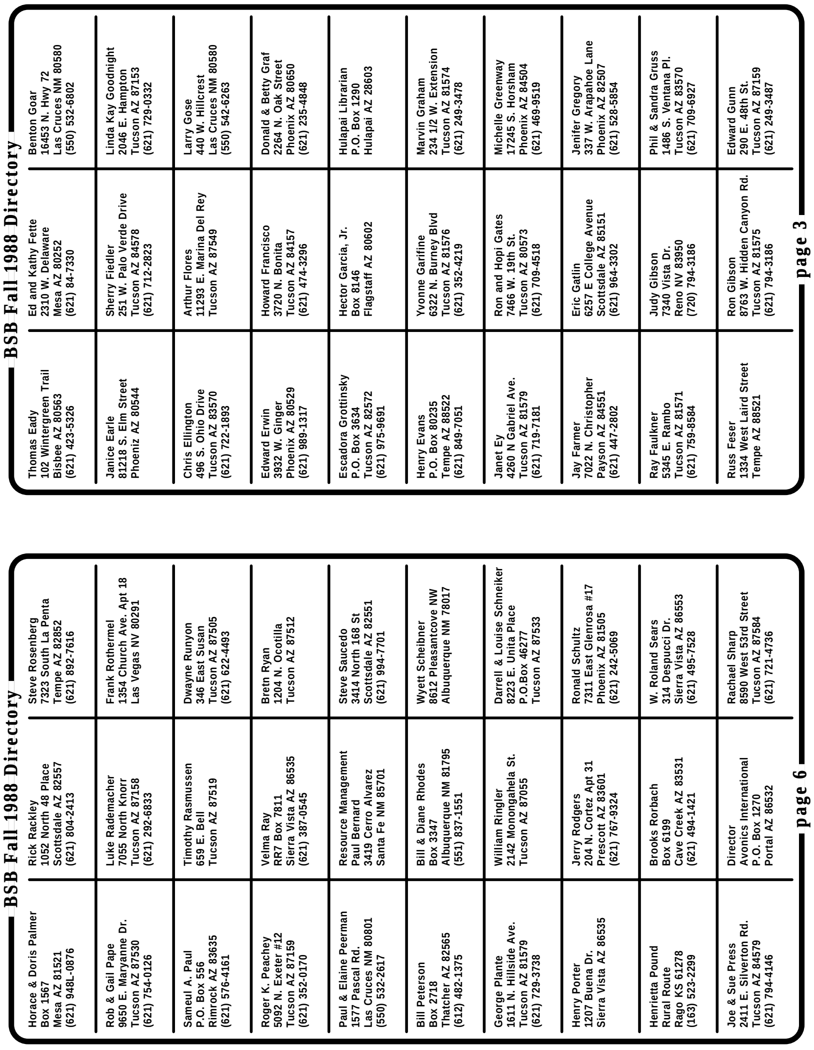
duper custom RISC computer chips to several manufacturers who are now offering all sorts of new LaserJet lid options, as well as "real" PostScript raster image processors that will plug directly into various personal computers. Contact *Weitek* directly for a current list of retail suppliers. Some of the Weitek implementations run as much as five times faster than the high end Apple LaserWriter NTX and do so at a potentially lower cost.

It's also possible to install a new LaserWriter computer directly into a HP LaserJet. A $10 set of easy plans is available from *Custom Technology*, while the LaserWriter boards are sometimes available through *Shreve Systems*. There's two problems with this route. The main LaserWriter computer boards are very difficult to find at reasonable prices, while the much smaller and also required interface connector card is just about impossible to pin down.

There are also some interesting and low cost software-only solutions to the LaserJet problem. One of these is the *Freedom of the Press*, while a second is *GoScript* from *LaserGo*.

All you helpline callers have been especially enthusiastic over *GoScript*. Besides driving a LaserJet, this can even give an *Epson* dot matrix printer some limited PostScript capabilities, and the cost is under $200.

Yes, most of my *PostScript Show and Tell* will run under *GoScript*.



BSB Fall 1988 Directory

Thomas Eady
102 Wintergreen Trail
Bisbee AZ 80563
(621) 423-5326

Janice Earle
81218 S. Elm Street
Phoeniz AZ 80544

Chris Ellington
496 S. Ohio Drive
Tucson AZ 83570
(621) 722-1893

Edward Erwin
3932 W. Ginger
Phoenix AZ 80529
(621) 989-1317

Escadora Grottinsky
P.O. Box 3634
Tucson AZ 82572
(621) 975-9691

Henry Evans
P.O. Box 80235
Tempe AZ 88522
(621) 849-7051

Janet Ey
4260 N Gabriel Ave.
Tucson AZ 81579
(621) 719-7181

Jay Farmer
7022 N. Christopher
Payson AZ 84551
(621) 447-2802

Ray Faulkner
5345 E. Rambo
Tucson AZ 81571
(621) 759-8584

Russ Feser
1334 West Laird Street
Tempe AZ 88521

Ed and Kathy Fette
2310 W. Delaware
Mesa AZ 80252
(621) 84-7330

Sherry Fiedler
251 W. Palo Verde Drive
Tucson AZ 84578
(621) 712-2823

Arthur Flores
11293 E. Marina Del Rey
Tucson AZ 87549

Howard Francisco
3720 N. Bonita
Tucson AZ 84157
(621) 474-3296

Hector Garcia, Jr.
Box 8146
Flagstaff AZ 80602

Yvonne Garifine
6322 N. Burney Blvd
Tucson AZ 81576
(621) 352-4219

Ron and Hopi Gates
7466 W. 19th St.
Tucson AZ 80573
(621) 709-4518

Eric Gatlin
6257 E College Avenue
Scottsdale AZ 85151
(621) 964-3302

Judy Gibson
7340 Vista Dr.
Reno NV 83950
(720) 794-3186

Ron Gibson
8763 W. Hidden Canyon Rd.
Tucson AZ 81575
(621) 794-3186

Benton Goar
16453 N. Hwy 72
Las Cruces NM 80580
(550) 532-6802

Linda Kay Goodnight
2046 E. Hampton
Tucson AZ 87153
(621) 729-0332

Larry Gose
440 W. Hillcrest
Las Cruces NM 80580
(550) 542-6263

Donald & Betty Graf
2264 N. Oak Street
Phoenix AZ 80650
(621) 235-4848

Hulapai Librarian
P.O. Box 1290
Hulapai AZ 28603

Marvin Graham
234 1/2 W. Extension
Tucson AZ 81574
(621) 249-3478

Michelle Greenway
17245 S. Horsham
Phoenix AZ 84504
(621) 469-9519

Jenifer Gregory
337 W. Arapahoe Lane
Phoenix AZ 82507
(621) 528-5854

Phil & Sandra Gruss
1486 S. Ventana Pl.
Tucson AZ 83570
(621) 709-6927

Edward Gunn
290 E. 48th St.
Tucson AZ 87159
(621) 249-3487

page 3

BSB Fall 1988 Directory

Horace & Doris Palmer
Box 1567
Mesa AZ 81521
(621) 948L-0876

Rob & Gail Pape
9650 E. Maryanne Dr.
Tucson AZ 87530
(621) 754-0126

Sameul A. Paul
P.O. Box 556
Rimrock AZ 83635
(621) 576-4161

Roger K. Peachey
5092 N. Exeter #12
Tucson AZ 87159
(621) 352-0170

Paul & Elaine Peerman
1577 Pascal Rd.
Las Cruces NM 80801
(550) 532-2617

Bill Peterson
Box 2718
Thatcher AZ 82565
(612) 482-1375

George Plante
1611 N. Hillside Ave.
Tucson AZ 81579
(621) 729-3738

Henry Porter
1207 Buena Dr.
Sierra Vista AZ 86535

Henrietta Pound
Rural Route
Rago KS 61278
(163) 523-2299

Joe & Sue Press
2411 E. Silverton Rd.
Tucson AZ 84579
(621) 794-4146

Rick Rackley
1052 North 48 Place
Scottsdale AZ 82557
(621) 804-2413

Luke Rademacher
7055 North Knorr
Tucson AZ 87158
(621) 292-6833

Timothy Rasmussen
659 E. Bell
Tucson AZ 87519

Velma Ray
RR7 Box 7811
Sierra Vista AZ 86535
(621) 387-0545

Resource Management
Paul Bernard
3419 Cerro Alvarez
Santa Fe NM 85701

Bill & Diane Rhodes
Box 3347
Albuquerque NM 81795
(551) 837-1551

William Ringler
2142 Monongahela St.
Tucson AZ 87055

Jerry Rodgers
204 N. Cortez Apt 31
Prescott AZ 83601
(621) 767-9324

Brooks Rorbach
Box 6199
Cave Creek AZ 83531
(621) 494-1421

Director
Avonics International
P.O. Box 1270
Portal AZ 86532

Steve Rosenberg
7323 South La Penta
Tempe AZ 82852
(621) 892-7616

Frank Rothermel
1354 Church Ave. Apt 18
Las Vegas NV 80291

Dwayne Runyon
346 East Susan
Tucson AZ 87505
(621) 622-4493

Bretn Ryan
1204 N. Ocotilla
Tucson AZ 87512

Steve Saucedo
3414 North 168 St
Scottsdale AZ 82551
(621) 994-7701

Wyett Scheibner
8612 Pleasantcove NW
Albuquerque NM 78017

Darrell & Louise Schneiker
8223 E. Unita Place
P.O.Box 46277
Tucson AZ 87533

Ronald Schultz
7311 East Glenrosa #17
Phoenix AZ 81505
(621) 242-5069

W. Roland Sears
314 Despucci Dr.
Sierra Vista AZ 86553
(621) 495-7528

Rachael Sharp
8590 West 53rd Street
Tucson AZ 87584
(621) 721-4736

page 6

Fig. 3 – A PostScript intelligent directory page pair.

The apparent problems with host-based software solutions are that they are very slow and that the PostScript emulation is partial at best. Many of these also do not support the downloadable genuine Adobe PostScript fonts that are crucial to serious high end desktop publishing uses.

Finally, rumor has it that HP is shortly coming out with a brand new duplex (double sided) laser printer that uses the super secret advanced *Canon NX* engine. HP would have to be absolute idiots to not offer factory stock PostScript in this machine.

Time will tell. Meanwhile, I have got this anthill if you need it for anything. Anything at all.

Tell me all About A New Franchising Opportunity

It is called *Book Publishing on Demand* and it goes like this: Picture a machine in your video rental store or convenience market, a four foot cube. In it are CD ROM PostScript data bases for 75 thousand popular books. The phone line that goes into the wall is connectable to 75 million more titles. Make a keyboard entry, and out pops a custom bound book, $6 for a paperback or $9 for hardbound. With, of course, the buyer's name in gold on the cover.

The advantages and economics to creating only books that are needed and wanted are overwhelming. There are no shipping charges, no distributors or multi-tier distribution, no shopworn or dog-eared volumes, no returns, no stocking decisions, no fighting for shelf space or visibility, and no remainders.

And no more of the IRS paying publishers to shred books like it now insists on doing. No inventory means no backlist tax.

All titles now instantly available. Anyplace, anytime. There's no such thing as an out-of-print book. Ever.

Because of the superb economics, authors could now actually be paid a living wage. Say 20 percent of the final selling price of all copies sold for a beginning author, 25 percent for a proven writer, and as much as 33 percent for a gonzo best seller. And payment could be an instant banking credit for the better selling authors. With returns and all those similar ripoffs virtually unheard of.

The quaint and arcane concept of "getting a manuscript accepted" now would cease to be meaningful, as the process would end up as complicated as qualifying yourself to buy a quart of milk. Any submission would be quickly scanned for legality, readability, and sanity. A small formatting fee would then be paid. Presto. You are in print. Forever.

No more publisher's committees sitting on your manuscript for fourteen months and then rejecting it because it was "not timely". First review copies could now be available within minutes. Or even less if the reviewers have their own machines.

Many pieces of book publishing on demand are rapidly falling in place. I am actually profitably doing a lot of this on my own, albeit on a manual, rather than an automated basis. All gets done "low end" with outstanding economics. And those helpline calls have created a network of associates that also have a very heavy interest in this exciting new opportunity.

Write or call if you want to join the team or get in on the network. No charge and no obligation, of course.

What is GOCCO?

Every month, I feel I just have to be getting to the bottom of the barrel of unique new LaserWriter and PostScript materials and opportunities. And every month, at the last possible instant, along comes some old product scunging away somewhere on a forgotten dusty shelf that, one more time, opens up great heaping gobs of incredibly mind-blowing new desktop and laser printing worlds.

This month, the product is called *GOCCO*. This system will let you instantly convert any laser printed image into a silk screen off of which you can do high quality "real" printing with "real" ink onto just about any surface imaginable. Even several colors at once if you want to.

The system, materials, and how-to books are available either from *Think Ink* or *Polycor*.

The system is similar to those old thermal transfer masters. Your silk screen is pre-coated with a thin and meltable wax. Light from several flashbulbs or an old thermal master machine shines through the screen and onto your laser print. The black toner parts get hot, melt, and pick up the wax, opening holes in the screen. The white parts stay cool and leave the wax on the screen.

You have a positive acting system, since black on the print makes holes in the screen which puts ink onto the final page or whatever it is you are printing on. Each screen is good for hundreds of impressions.

```
%  Copyright c 1989 by Don Lancaster and Synergetics, Box 809, Thatcher, AZ 85552.
%  (602) 428-4073. All rights reserved. Personal, non-commercial use permitted so long as
%  this header remains present and intact. Latest work-in-progress disk costs $39.50.

%  Requires my gonzo justify and my ps.util.1 Free printed copies on request.

ps.util.1 begin printerror nuisance begin landscape

/font0 {/NewCenturySchlbk-Bold findfont [1.2 0 0 1.4 0 0] makefont setfont} def
/font1 {/Helvetica-Bold findfont [0.7 0 0 0.7 0 0] makefont setfont} def
/font2 {/Helvetica findfont [0.8 0 0 0.8 0 0] makefont setfont} def

/namesperpage 30 def /colcheck {} def

/sheet81 {/leftpagenum (page 8) def /rightpagenum (page 1) def /leftstartingname 7
namesperpage mul def /rightstartingname 0 namesperpage mul def} def

/sheet27 {/leftpagenum (page 2) def /rightpagenum (page 7) def /leftstartingname 1
namesperpage mul def /rightstartingname 6 namesperpage mul def} def

/sheet63 {/leftpagenum (page 6) def /rightpagenum (page 3) def /leftstartingname 5
namesperpage mul def /rightstartingname 2 namesperpage mul def} def

/sheet45 {/leftpagenum (page 4) def /rightpagenum (page 5) def /leftstartingname 3
namesperpage mul def /rightstartingname 4 namesperpage mul def} def

/printnames {/startwith exch def /xpos 1 def /ytop 55.2 def /ypos0 ytop def /ypos ypos0
def /yinc 0.9 def /yshift 56 10 div def 0 0 (\0332)cc 3 {10 {xpos ypos brbarray startwith
get cl /ypos0 ypos0 yshift sub def /ypos ypos0 def /startwith startwith 1 add def} repeat /ypos0
ytop def /ypos ypos0 def /xpos xpos 35 3 div add def} repeat} def

/background {25 21 10 setgrid bestgray lightgray 0.7 setgray 0 35 57 57 1.2 0.4 quickboxdraw
white 9 57 mt 17 r 15 0 mt 5 r black 35 2 div 56.6 (\0330BSB Fall 1988 Directory) cc gsave
bestgray 0.7 setgray  black 1 setlinecap 0.2 setlinewidth [{35 3 div 0.7 mt 55 u} 35 3 div 2 ]
xrpt [{0.7 56 10 div 0.5 add mt 33.6 r} 56 10 div 9 ] yrpt grestore } def

/generatethepage {createarray save /leftsnap exch def background 35 2 div -0.3 leftpagenum
cc leftstartingname printnames clear leftsnap restore save /rightsnap exch def 11 72 mul
2 div 0 translate background 35 2 div -0.3 rightpagenum cc rightstartingname printnames clear
rightsnap restore} def

/delimiter (
%%%%%
) def

/createarray { 1000 array /brbarray exch def /pointer 0 def 0 1 999 {brbarray exch ( ) put} for
brblist {delimiter search {brbarray exch pointer exch put /pointer pointer 1 add def pop}
{exit} ifelse} loop} def

/brblist
(Aaron A. Aardvark
1 First Avenue
Alton, AL 32768
(101) 111-1101
 . AM82 M84 Exp 1/1/90
%%%%%
Byron B. Benson
2 Boyd Boulevard
Bismark CA, 22021
(202) 222-2202
 . M87 88 Dues not paid
%%%%%

(This continues for up to 240 names)

%%%%%
Zack Zimmerman
99999 Zoraster Trail
Zion, UT 99909
(999) 999-9909
 .FM86 M88 Exp 1/1/94
%%%%%
%%%%%) def

% selections here are sheet81, sheet27, sheet63, or sheet45

sheet63   320 copies   generatethepage   showpage
```

Fig. 4 – PostScript code for the intelligent directory.

Mucho thanks to that absolutely outstanding *Whole Earth Review* for putting me onto this great product, as they do dozens of other times each issue. And thanks to Don Harbolt, a winner in our color proofing contest, for his valuable input on this.

An afterthought: Could GOCCO be modified so that, instead of using a wax, some more permanant thermographic material that might produce "raised print" on your laser output? Perhaps two components that, when melted together, can create a durable varnish or raised epoxy. That would really be potent.

In fact, any way of handling laser thermography would be real handy.

```
% Copyright c 1989 by Don Lancaster and Synergetics, Box 809, Thatcher, AZ 85552.
% (602) 428-4073. All rights reserved. Personal, non-commercial use permitted so long as
% this header remains present and intact. Latest work-in-progress disk costs $39.50.

% Requires my gonzo justify and my ps.util.1 Free printed copies on request.

ps.util.1 begin printerror nuisance begin landscape

/font0 {/NewCenturySchlbk-Bold findfont [1.5 0 0 1.5 0 0] makefont setfont} def
/font1 {/Helvetica-Bold findfont [0.7 0 0 0.7 0 0] makefont setfont} def
/font2 {/Helvetica findfont [0.8 0 0 0.8 0 0] makefont setfont} def

/colcheck {} def /startwith 0 def /showthegrids false def

/printnames { /yinc 1.8 def 0 0 (\0330)cc gsave  20 22 translate 90 rotate #ofnamestoprint {0
0 brbarray startwith get cl copypage newpath -2 -10 mt 15 pu 30 pr 15 pd closepath gsave 1
setgray fill grestore /startwith startwith 1 add def} repeat grestore } def

/logo {  % your return address or return logo proc goes here
} def

/background {25 21 10 setgrid showthegrids {35 57 showgrid} if bestgray lightgray 0.7
setgray 0 35 57 57 1.2 0.4 quickboxdraw black newpath logo} def

/generatethepage {/yinc 2 def save /rightsnap exch def 11 2 div 72 mul 8.5 72 mul translate
180 rotate 25 21 10 setgrid  showthegrids {35 57 showgrid} if bestgray lightgray 0.7 setgray
4.5 26 40 11 1.2 0.4 quickboxdraw black /yinc 2 def 17.5 36 blurbtext cc clear rightsnap
restore createarray save /leftsnap exch def 11 72 mul 2 div 0 translate background
printnames clear leftsnap restore} def

/delimiter (
%%%%%
) def

/createarray { 1000 array /brbarray exch def /pointer 0 def 0 1 999 {brbarray exch ( ) put} for
brblist { delimiter search {brbarray exch pointer exch put /pointer pointer 1 add def pop}{exit}
ifelse} loop} def

/blurbtext
(\0330Your Fall 1988 B.R.B.
Directory, Meeting Notice,
And Call For Papers) def

/brblist
(Aaron A. Aardvark
1 First Avenue
Alton, AL 32768
(101) 111-1101
 . AM82 M84 Exp 1/1/90
%%%%%
Byron B. Benson
2 Boyd Boulevard
Bismark CA, 22021
(202) 222-2202
 . M87 88 Dues not paid
%%%%%

(This continues for up to 240 names)

%%%%%
Zack Zimmerman
99999 Zoraster Trail
Zion, UT 99909
(999) 999-9909
 .FM86 M88 Exp 1/1/94
%%%%%
%%%%%) def

% add handfeed here if wanted

/#ofnamestoprint 300 def   generatethepage
```

Fig. 5 – Intelligent Directory Auto-Addressing Cover.

How Do I get Started Using PostScript?

As should be ridiculously obvious by now, PostScript is *the* best way of combining text and graphics into useful page descriptions for laser printers and phototypesetters.

Back to all of the fundamentals. PostScript is a universal and general purpose computer language that is distantly related to *Forth*. It excels at graphics and text page image buildup through its device independence; its outstanding curve drawing abilities; its powerful techniques for translation, rotation, scaling, and for other transformations; and for its extensive font machinery.

Unlike several lower level printer languages, a single PostScript font can be shown in any size from 2 points to 60,000 points in quarter point increments, and shown in any direction on the page, or along a circular or even totally arbitrary path. Powerful post-path techniques also give you a virtually infinite number of variations upon any of the many thousands of PostScript fonts that are available today.

To use PostScript, your printer or output device *must* have a built-in PostScript interpreter. As a user, you can crete your PostScript programs using nothing but words and numbers in any old word processor. Or, you can use any of dozens of popular PostScript speaking canned applications programs. Or any of a number of powerful emulators that temporarily downgrade PostScript so as to get it to behave like older printer and plotter languages.

Because of its nearly total device independence, PostScript will work with virtually any computer, although many of the more powerful canned applications programs tend to be mostly Mac and IBM oriented. On the other hand, I have yet to find *any* program on *any* computer that is remotely as good as AppleWriter on a IIe when it comes to quickly and conveniently developing custom or unique PostScript code.

PostScript today is aimed mostly at laser printers and phototypesetters. Software is now becoming available that lets you run PostScript on low end printers, for plotters, engraving machines, and even fax systems. And it is only a matter of time that display PostScript will become a de-facto graphics standard, owing both to its device independence and raw power.

If you think about it for a while, it sure seems downright stupid to have one screen graphics standard, and a second printer standard. As the Mac people have found out the hard way, QuickDraw simply cannot hold a candle to display PostScript when it comes to rich and powerful page descriptions.

So how do you get started using and learning PostScript on your own?

If you do not already have access to a PostScript speaking printer, start with my *Intro to PostScript* video, these *Ask the Guru* reprints, or that blue *PostScript Tutorial and Cookbook* from Adobe Systems.

Later on, when you get your own printer, pick up my *PostScript Show and Tell* disks, available for most popular personal computer brands, and purchase Adobe's red *PostScript Reference Manual* and their green *PostScript Program Design* books. I do try to stock all these.

Or write and call me for more specific suggestions. We now get 30 to 50 PostScript helpline questions per day. Best calling times are 8-5 weekdays, mountain standard time.

What is This Month's PostScript Utility?

Helpline callers often ask me why they should bother learning and using "raw" PostScript when there are all of those canned page making and illustration applications programs that are just sitting out there.

Well, first, PostScript is lots of fun and is insanely addictive. That should be enough of an answer right there. At least for you new age types.

Second, your own PostScript code can often run ridiculously faster than all the canned stuff. A three column, 6000 character gonzo justified layout with two figures, a header, and a footer should only take three seconds or so above the normal paper feeding speed when you use your own code.

Third, you can often get a higher quality final printed result, especially when it comes to such things as text justification, a decent looking gray, smooth curves, unusual borders, or effective step-and-repeat procs.

And, fourth, using raw PostScript lets you do things in your own way, rather than by using stuff that someone else decided just absolutely had to be good for you.

But far and away the most compelling reason to learn and use raw PostScript is that it often lets you easily and quickly do things that would seem impossible or extremely difficult to do with those canned applications programs.

Let's look at printing an intelligent directory as an example. But, before we do, and to put my money where my mouth is, let's have another one of our stupid contests.

Just show me *any way at all* to use a canned pagemaking or illustration applications package to do what follows that is even remotely as fast and as convenient as using raw PostScript and AppleWriter on a IIe, and you'll win an *Incredible Secret Money Machine*. In addition, the best entry of all will win an all expense paid (FOB Thatcher, AZ) *tinaja quest* for two.

Back to the program. Our goal here is to quickly and conveniently publish a directory for a club or a medium sized organization of 100 to 1200 members. The directory is to be a folded booklet, and all the directory entries are to be in an "open" form that is easy to use and easy to read. It should also be possible to chop up a directory and use it as quick response mailing labels.

The directory must accept input from any word processor or data base in the world, and all directory entries should fully support some "hidden" and non-printing information such as their membership class, when they joined, their membership numbers, etc. The directory should also be able to internally self-address and self-mail to its own list, doing so at the full "wide open" printer speed.

Above all, the directory has to be smart enough that it will automatically put, say, page eight to the left of page one on the same landscape sheet, and automatically put the right names in the right order on each and every page. As names get added or removed anywhere in the list, all the others will automatically get shoved around each page or between pages or columns as needed.

Figure three shows you a typical directory page before folding, while the PostScript code for the internal pages is shown in figure four.

Most of the self-addressing and auto-mailing Post-Script cover code appears in figure five. Note that the addressing gets done at the same time the rest of the cover is printed.

This does require my gonzo justify stuff we looked at in the October 1988 *Computer Shopper*. Write or call for a free printed listing. This is also available ready to run on my *Work in Progress* disk and in *Ask The Guru*, volume II.

Each sub-page uses a gray border, with the organization name at the top and the page number at the bottom.

The input list is scanned between selected markers, which can let you have three, four, or even five lines in the address without any hassles. In this example, five percent signs are used as the delimiting marker. You can substitute any delimiter you like.

It is up to your word processor or whatever to select the printing versus non-printing lines. In the case of AppleWriter, any data line that starts with a period gets ignored. This lets you put hidden lines into each entry. To drop out the phone number on the actual addressing, you can either prefix your number with a period, or else search and replace the list and eliminate any line that starts with an opening parenthesis.

The initial list then gets converted to an array of addresses. This greatly simplifies finding the initial address for each sub-page.

A sneaky trick is used to do the self-addressing. Instead of creating your entire mailing cover art over and over again, you simply will *erase each old name and put a new one down*. This is insanely faster and runs at the full speed of just about any PostScript printer.

As shown, the code is specific for an eight page directory of up to 240 names. It can be easily customized for other lengths, or even fully automated to produce whatever number of pages the list demands.

To also give you a chance at a winnable contest, just show me any variation at all on the theme of intelligent and self-mailing directories. Let me hear from you.

Don Lancaster's

ASK THE GURU

January, 1989

PC Transporter first look
PostScript intelligent tabs
LaserWriter NTX defects list
Simple white noise generator
Printer and printing resources

What is the future of the Apple IIgs? Stan Veit and I have been kicking this around, and we are getting a few different answers from several different sources.

At present, the unit sales of the IIgs are pretty near the same as those of the IBM XT, at four percent of the total computer market. These figures are from *Computer Reseller*, a pretty much pro-IBM trade journal that is a great way to keep score of just who is doing what to whom.

Apple is sending you conflicting signals. They have just upgraded the IIgs operating system and are now agressively hiring a big bunch of new top-quality Apple II people. Apple IIc, IIe, and IIgs end user and developer support is far and away the best it has ever been. The same goes for Apple's own publications and tech info. Those new "big machine" products from third parties, such as the improved *AppleWorks*, *PC Transporter* and the great new GEOS graphics environment are coming on-line and making a big splash.

On the other hand, Apple has been witholding the IIgs upgrade, and they purposely crippled that recent IIc upgrade by keeping AppleTalk and a real time clock off it. And the introduction of a Mac K-12 is imminent.

As I see it, there is only one fatal flaw in the IIgs. And that lies in all of the monumental costs, the incredible time delays, rude surprises, and the inexcusable frustration levels that are involved in that APW development environment. If you have to go to all that innane nonsense, then you might as well be doing it all on a Mac. At present, any serious IIgs commercial software development appears to be a sucker bet because of the sheer frustration problems involving APW.

The long term handwriting is on the wall. Apple now has a secret new machine in the works known as the *Brooklyn Bridge* that runs both Mac and IIgs software side by side.

Actually, if you think about it for a while, the day of the custom operating system is nearing an end. Instead, you just take lots of RAM and a RISC chip that's running like a bat out of Cupertino, and then provide suitable firmware microcode that temporarily downgrades it into your choice of a 80386, a 65832, or a 68030. And then runs anything by anybody. Real time or even faster.

Similarly, disk drives are getting smarter and more flexible. A "multisync" drive that can accept any past or present media shouldn't be all that far away. Interestingly enough, the new GS/OS operating system does include a FST, or *File System Translator* code that, in theory, will let you read from or to any disk file in any format, crossing any boundary.

Oh yeah. There is a stupidity on the new IIgs version 4.0 GS/OS operating system that is driving all you newcomers up the wall. At a first glance, it appears as if there is no support whatsoever for 5-1/4 inch disk drives in this new operating system. Actually, the needed 5-1/4 drivers are buried in a subdirectory named /TOOL.DIRECTORY. A copy of the drivers have to be moved up into /SYSTEM/SYSTEM.DRIVER/.

(1) On an Apple II+, IIc, IIe, or IIgs, get into **BASIC.SYSTEM** and do a **CALL -151** to get into the monitor.

(2) Then, enter the following code:

```
0300:  4C 07 03 27 0C A1 C9 AD  <cr>
0308:  06 03 0A 0A 0A 4D 06 03  <cr>
0310:  0A 0A 90 03 2C 30 C0 2E  <cr>
0318:  03 03 2E 04 03 2E 05 03  <cr>
0320:  2E 06 03 4C 07 03 11 11  <cr>
```

(3) Finally, do a **BSAVE KFC.VIRUS, A$0300, L$28, D2**

(4) To test, use, or abuse your code, do a **BRUN KFC.VIRUS**

Fig. 1 – A simple "white noise" generator.

Full tech details on GS/OS are now available from *APDA*.

Meanwhile, *Hewlett-Packard* has finally gotten around to paying their PostScript license fees. No sense in rushing these things. Another very few years, and it should actually be possible to do useful desktop publishing from a new factory stock H-P LaserJet printer.

My views on what's-his-name's NeXT machine? Thought you would never ask. It does clearly show its Lisa and Apple III heritage. And it definitely sews up the "CD audio pirate who is also a Shakespeare loving university professor" market.

There's a lot of similarities here with the Amiga, which also was a machine that introduced some new innovations a few weeks ahead of their time that all the others promptly stole and ran with. This time, the key developments include display PostScript, fully professional audio, and an erasable and removable CD-ROM disk system. Fatal flaws currently include the lack of ordinary floppy disk drives, lack of color, and the market focus that is far too narrow.

Apple has now released their new *LaserWriter Reference Manual*. Included are complete lists of all the *Diablo* and the *LaserJet* commands, secrets of hard disk access, interface details, the works. I do have a few copies in stock here at *Synergetics* if you need one.

Speaking of which, we do have autographed copies of volume I and volume II of *Ask the Guru* in stock now, as well as volume II of all my *Hardware Hacker* stuff. And, if you need the very latest and the very best of all my PostScript goodies, do look into my *Work in Progress* disks.

As per usual, this is your column

and you can get technical help and off-the-wall networking per the end box that follows the usual *Names and Numbers* section.

Let us start off by giving you a bunch of static . . .

Show me a Simple White Noise Generator

Few people realize how easy it is to generate white noise sound effects on most any Apple. Figure one gives you a short machine language code routine you might like to try.

This is based on a 31 stage pseudo random sequence generator. It is just a long shift register with feedback from stages 28 and 31 being EOR'ed together to form the next input. After each shift, out comes an *apparently* random 1 or 0. While the ones and zeros are apparently random and do in fact obey most of the rules of random numbers, that same sequence will repeat over and over again, once each 2,147,483,647 counts.

If you get a one out, you whap the speaker, while if you get a zero out, you do nothing. The results sound like your Apple is frying itself in its own grease, and it is a pretty good (although slightly pinkish) approximation to a white noise source.

The code might be extended for various sound effects. It can also become the basis for a highly useful random number generator that is far better than the fatally flawed one that is used in AppleSloth.

More details on all this, with full working source and object code, do appear in my *Assembly Cookbook For The Apple II and IIe*. Write or call if you need an autographed copy.

What is The PC Transporter?

I am not a great fan of *MS/DOS*. I feel it is insipid, stilted, and grossly uninspiring. Totally lacking in vibes of any nature. The sort of thing you would expect a middle management insurance actuary to be using while working on the fifteenth floor of a Milwaukee office building.

Nonetheless, there sure are a lot of people laboring under the delusion that they are willing to actually *pay* for disks that say *MS/DOS* on them.

So, since I am a card carrying and graduate member of the P.T.Barnum school of marketing theory, all of my software is initially developed on an Apple IIe or IIgs and then translated and sold to unsuspecting IBM, Mac, and Atari folks.

As you can imagine, it has been a real problem to translate and then duplicate disks. In particular, all of the *MS/DOS* translations did involve hauling Bee's IIc across town and connecting it to a Hong Kong clone, sending the files over with Apple-Writer and receiving them with *PC Talk*. But Kate moved, and so did her Hong Kong machine. What to do?

It was *Applied Engineering* to the rescue. They have a nifty product called the *PC Transporter*, which is a plug in card for a IIe or IIgs that is an entire American made Hong Kong clone on a card, except that it runs three times faster than the real thing.

There is now a *MS/DOS* drive on my Apple, and you simply use a copy command to translate PostScript files or whatever else you care to between the two disk standards.

I've only had the beast for a few days, but the high quality engineering that went into it is obvious. It is even priced less than a stand-alone clone would be. And the *PC Transporter*

1. The per-page toner costs are currently as much as 15 times higher than on the earlier LaserWriter printers, making the NTX totally useless for serious production work.
2. The factory toner is excessively abrasive and the cartridge drums are far too scratch prone. Edge shading effects are also unacceptable.
3. The lack of user toner refillability on factory stock cartridges is an outright and intolerable atrocity.
4. Current Apple "black hole" service policies demand that you must use Hewlett Packard repair manuals and H-P repair parts.
5. Any custom serial interface settings get totally trashed if any use at all is ever made of AppleTalk through switch changes.
6. The mode changing switches are far too flimsy.
7. The lack of any handles makes the NTX extremely easy to drop. This is an acute problem for demos, lectures, and school use.
8. The backwards numbering on the print density dial is just plain wrong. It should also be outside adjustable.
9. The paper tray is still far too shallow. One ream plus a hundred pages is the absolute minimum.
10. Envelope print quality, while greatly improved, is still marginal.
11. The PostScript **frametofile** command is conspicuously absent.
12. Many PostScript functions, notably **eexec**, **cexec**, **superexec**, and **FlxProc**, among others, are improperly documented.
13. The font lockout on **pathforall** is a second inexcusable atrocity.
14. There is no video output, nor any way of accessing final bitmaps.
15. Hard disk usage is unnecessarily restrictive.
16. AppleTalk is often ridiculously slower than standard serial access.
17. Recent speedup techniques, including the 68030, the Weitek RISC chip, and new firmware leave the NTX as a rather slow machine.
18. The $150 so-called "required AppleTalk" cables can be replaced with an ordinary $5 printer cable for many users much of the time.
19. AppleTalk is not device independent; it treats newline characters different than does the serial comm channels.
20. The NTX costs too much.

Fig. 2 – Defects in the LaserWriter NTX.

certainly does a lot more than the limited demands I've asked of it.

So, I'll hold off on giving you a thorough review just yet. The beast works and works well and solves a lot of problems for me. Nuff said.

Don't miss the egress. This way to the egress . . .

What's Wrong With the NTX?

Well, I have had my *LaserWriter* NTX for eight months now, and I do suppose it is time to tell you what is wrong with it. I have now thoroughly tested it out on a twelve hour per day basis, doing all my columns, walk-in typesetting, ad materials, my internal forms, and even attempting book-on-demand publishing with it.

First and foremost, the *Canon SX* engine is far and away the finest and most popular "low end" laser printing engine. And the NTX is the finest SX PostScript implementation available to date if you ignore all the costs.

So, *if* you have access to an Apple educational or developer discount, or *if* you are spending someone else's money, then the NTX is far and away the one to buy. Otherwise, that *NEC LC-890* represents a very good, and very economical choice.

Incidentally, the NeXT hype about their $2000 PostScript printer is just that. To build a NeXT printer, you just take an already brain dead $1400 street price LaserJet and then *remove* some electronic components from it. When you have display PostScript in your host, you can get away with use of a brain dead engine. So long as you don't mind tying up your host.

Anyway, on to the big NTX roast. Figure two shows you some of the main defects in this machine that I have run across. You can judge how important these are to yourself.

I feel that the real NTX killer is the 15:1 difference in the per-page toner costs between it and all the earlier LaserWriters. This effectively locks out the NTX from virtually all production uses, and has forced me to drop back to using my 400,000 copy LaserWriter Plus for my book-on-demand production printing.

How does this 15:1 cost penalty come about? On the LaserWriter Plus, you can buy used cartridges for as low as $5, and then refill them up to seven times with third-party toner for as little as $7.50 per shot. This can give you per-page toner costs in the 0.33 cents per page range. And, yes, those images are nearly as black when you do use a good third-party toner source. The cartridges get up to their blackest after their second or third toner refill.

On the LaserWriter NTX, it is hard to find any cartridges under $115, and they usually will scratch before you can even refill them once. This gives you a nickel per page or higher toner cost, and the resultant 15:1 penalty.

Interestingly, if you *immediately* remove that highly abrasive *Canon* factory toner from a new cartridge and refill it with a good third party toner, then you can get several refills. Give the factory toner to your neighborhood diesel mechanic for use as valve grinding compound.

If the NTX if to survive at all, the factory cartridges simply *must* get

American Printer
300 West Adams
Chicago, IL 60606
(312) 726-2802

Business Forms & Systems
401 North Broad Street
Philadelphia, PA 19108
(215) 238-5300

Canadian Printer & Publisher
777 Bay Street
Toronto, ONT M5W 1A7
(416) 596-5781

Colophon
1585 Charleston Road
Mountain View, CA 94039
(800) 833-6687

Direct Image Corp
1350 S Monterey Pass Road
Monterey Park, CA 91754
(213) 264-2000

Federal Graphics
120 Willow Stret
North Andover, MA 01845
(508) 681-8578

Font & Function
PO Box 7900
Mountain View, CA 94039
(800) 833-6687

Form
433 East Monroe Avenue
Alexandria, VA 22301
(703) 836-6232

Forms Professional
401 North Broad Street
Philadelphia, PA 19108
(215) 238-5300

Graphic Arts Abstracts
4615 Forbes Avenue
Pittsburgh, PA 15213
(412) 621-6941

Graphic Art Lit. Abstracts
One Lomb Memorial Drive
Rochester, NY 14623
(716) 475-2549

Graphic Arts Monthly
875 Third Avenue
New York City, NY 10022
(212) 605-9400

Graphic Arts Product News
29 North Wacker Drive
Chicago, IL 60606
(312) 726-2802

Graphic International
PO Box 4639
Pompano Beach, FL 33063
(305) 971-4360

High Volume Printing
Box 368
Northbrook IL, 60062
(312) 564-5940

Image World RIT
One Lomb Memorial Drive
Rochester, NY 14623
(716) 475-2549

In-Plant Printer
Box 368
Northbrook, IL 60062
(312) 564-5940

In Plant Reproductions
401 North Broad Street
Philadelphia, PA 19108
(215) 238-5300

Instant Printer
Box 368
Northbrook, IL 60062
(312) 564-5940

Modern Office Technology
1100 Superior Avenue
Cleveland, OH 44114
(216) 696-7000

NAPL
780 Palisade Avenue
Teaneck, NJ 07666
(201) 342-0700

Office Systems
941 Danbury Road
Georgetown, CT 06829
(203) 544-9526

Package Printing
401 North Broad Street
Philadelphia, PA 19108
(215) 238-5300

Paper, Film & Foil Converter
29 North Wacker Drive
Chicago, IL 60606
(312) 726-2802

Plan & Print
9931 Franklin Avenue
Franklin Park, IL 60131
(312) 671-5356

Print
355 Lexington Avenue
New York City, NY 10017
(212) 682-0830

Print Equipment News
Box 5540
Glendale, CA 91201
(818) 954-9495

The Printers Shopper
PO Drawer 1056
Chula Vista, CA 92012
(800) 854-2911

Printing Impressions
401 North Broad Street
Philadelphia, PA 19108
(215) 238-5300

Printing Journal
Box 5515
Pasadena, CA 91107
(818) 793-7901

Printing News
468 Park Avenue South
New York City, NY 10016
(212) 689-9690

Publishing Technology
401 North Broad Street
Philadelphia, PA 19108
(215) 238-5300

Quick Printing
1680 SW Bayshore Blvd
Port St Lucie, FL 34984
(407) 879-6666

Screen Printing
407 Gilbert Avenue
Cincinatti, OH 45202
(513) 421-2050

Southern Graphics
410 Verona Street
Kissimmee, FL 32742
(305) 846-2880

Systemdict
PO Box 7900
Mountain View, CA 94039
(800) 833-6687

Target Marketing
401 North Broad Street
Philadelphia, PA 19108
(215) 238-5300

Type World
15 Oakridge Circle
Wilmington, MA 01887
(617) 658-6876

U&lc
2 Hammarskjold Plaza
New York, NY 10017
(212) 371-0699

Fig. 3 – Some printing and printshop resources.

lowered in price; *must* be made far more scratch resistant; *must* internally provide non-abrasive toner; and absolutely, positively, and emphatically *must* be made easier to refill by the end user.

Show Me Some Print Shop Resources

All those of you that are already following my *Hardware Hacker* column over in *Radio-Electronics* are painfully aware that I am a great fan of trade journals. I personally subscribe to nearly 500 of these.

Trade journals are the controlled circulation "insider" magazines that do let you zero in on products and secrets of most any field of endevor.

You normally get a free subscription to a trade journal by filling out a qualification card that will tell them what they want to hear. It is also possible to gain a free subscription just by asking for an advertising rate card and requesting a sample copy. Additional ploys are included in my *Incredible Secret Money Machine*.

The reference shelf at your local library should hold both *Uhlricht's Periodicals Dictionary* and that *International Standard Periodicals Dictionary*. Together, they list some 45,000 or so trade journals and other magazines available today. Of these two, I prefer Uhlricht's.

At any rate, if you are going to get at all serious about desktop publishing, sooner or later you'll be wondering what the "big boys" are up to. Figure three lists some traditional print shop and printing resources that you might find of more than passing interest. While most of these are free trade journals, a few supply houses, product catalogs, trade associations, and fee-based mags are included.

Many thanks go to John Henry of Mitchell Printing for his help on this. John was also the *tinaja quest* winner in our color proofing contest.

Please let me know of anything else you feel should be included in this list. We'll even make a response one way to win this month's contest.

What is This Month's PostScript Utility?

I've been doing a lot of custom PostScript programming lately that seems to share one common thread. There are an incredible number of you out there that want to take data from any old data base off any old computer and attractively convert it into PostScript tables that use proportional spacing and mixed fonts, seperately being able to selectively use left, center, right, or fill justification on each individual column.

All bounded with fancy PostScript line art. And, of course, all done with perfect column alignment. And all done with a minimum of new typing.

The key problem is that most older data bases assume that you are using fixed pitch printer fonts. As soon as you go to a second column, you get a ragged column alignment and other nasty problems.

Well, figure four shows you my new PostScript *ctab* (callout tabular) routine. It will semi-automatically accept input from any data base and convert it into any number of columns, each column of which can be independently positionable, in any

```
% Copyright c 1989 by Don Lancaster and Synergetics, Box 809,
% Thatcher, AZ 85552. (602) 428-4073. All rights reserved. Personal
% non-commercial use permitted so long as this header remains present
% and intact. The Latest work-in-progress disk costs $39.50.

% This is intended as an add-on for version 6.0 or higher of my gonzo
% justify code but it can be adapted to other layout or formatting code.

% . . . . . . . . .

% ctab justify - will take most any input file and convert it into a
% proportionally spaced table. The only current rule is that double spaces
% are prohibited in any entry and that at least two spaces are needed
% between successive columns of the input file.

% tab list dictionary entries are -xoffset- -{justification}- -(fontnumber)-
%  -txtwide-. The width is always required but is normally only used with
% fill justify. It must be larger than any left justified string, unless you
% purposely want multiple line entries.

/tablist 100 dict def tablist begin

/0 [  0 {cl} (1) 1000] def
/1 [ 70 {cc} (1) 1000] def
/2 [150 {cf} (1)   80] def
/3 [250 {cr} (1) 1000] def
/4 [250 {cl} (1) 1000] def
/5 [300 {cl} (1) 1000] def
/6 [350 {cl} (1) 1000] def
/7 [400 {cl} (1) 1000] def
/8 [450 {cl} (1) 1000] def
/9 [500 {cl} (1) 1000] def
end

/tabstring (  ) def % this is the searching tab marker (two spaces)

/settabstring {save /snapct exch def /justifylastparline true def tablist
tabnum (   ) cvs get aload pop /txtwide exch def 0 get changefont exch xleft
add ypos 4 2 roll cvx exec clear snapct restore unrestorefont} def

/gottatab {exch pop exch /smsg exch def settabstring {smsg ( )
anchorsearch {pop /smsg exch def}{ exit} ifelse } loop /tmsg exch def } def

/tabscan {/tmsg exch def tmsg (\033h) search {/halfline true def}{/halfline
false def} ifelse /tabnum 0 def {tmsg tabstring search {gottatab /tabnum
tabnum 1 add def}{settabstring exit} ifelse} loop /ypos ypos yinc halfline
{0.5 mul} if sub def} def

/ctab {/ttmsg exch def /ypos exch def /xleft exch def /tabnum 0 def {ttmsg
(\n) search {exch pop exch /ttmsg exch def tabscan} {tabscan exit} ifelse}
loop clear} def
```

Fig. 4 – PostScript code for intelligent data base tabbing.

number of fonts you like, and in any justification mode of any width.

The code is intended for use with all my gonzo justify routines that we did see back in the October issue, but you can adapt it for any code. A *ctab* example appears in figure five.

You first set up an array that gives you the offset, the justification mode, the initial font, and the total allowable width for each column. It pays to have at least one extra entry in this array, just in case there might be some extra trailing spaces present in your data base code.

Note that this is a mixed array. The -x- offset is a decimal value. The justification mode is a proc, bracketed by curly braces. The initial font selection is a string numeric, while the maximum column width is once again a decimal value.

```
%  input gonzo.6, ctab, and this code . . .

/font0 {/Times-Roman findfont [12 0 0 12 0 0] makefont setfont} def
/font1 {/Times-Bold findfont [12 0 0 12 0 0] makefont setfont} def
/font2 {/Times-Italic findfont [12 0 0 12 0 0] makefont setfont} def
/font3 {/Helvetica findfont [12 0 0 12 0 0] makefont setfont} def

tablist begin
/0 [  0 {cl} (1) 1000] def     % first column bold left justify
/1 [ 67 {cf} (0)   39] def     % second column normal fill justify
/2 [152 {cc} (2) 1000] def     % third column italic center justify
/3 [250 {cr} (0) 1000] def     % fourth column normal right justify
/4 [999 {cl} (0) 1000] def     % spare to swallow any trailing zeros
end

/yinc 13 def                   % vertical spacing
/maxsstretch 0 def             % a sneaky trick to stretch characters
/maxcstretch 100 def           % and not spaces for this fill justify

85 90 moveto 0 128 rlineto 278 0 rlineto 0 -128 rlineto closepath stroke

% . . . and this data base info . . .

     100 200
     (
     aardvark   sulphur    eggplant    $12,345.06
     giraffe    oxygen     artichoke     $1297.55
     gnu        silicon    okra             $3.22
     javelina   krypton    brocolli       $345.96
     \033h
     elephant   sodium     rhubarb     $36,834.71
     wallaby    carbon     celery           $1.23
     ostrich    helium     potato         $223.41
     kangaroo   gallium    cucumber      $1254.33
     ) ctab
     showpage

%  . . . to get this result . . . .
```

aardvark	sulphur	*eggplant*	$12,345.06
giraffe	oxygen	*artichoke*	$1297.55
gnu	silicon	*okra*	$3.22
javelina	krypton	*brocolli*	$345.96
elephant	sodium	*rhubarb*	$36,834.71
wallaby	carbon	*celery*	$1.23
ostrich	helium	*potato*	$223.41
kangaroo	gallium	*cucumber*	$1254.33

Fig. 5 – Data base tabbing use example.

You also have to select a rule that tells you when to go from column to column. While the embedded tab could be used, I prefer using this pair of rules: "Any time you reach two or more sequential spaces, then a column change should take place." And: "Any two or more spaces in sequence are prohibited inside any single column entry."

The ctab code initially will isolate everything up to the first carriage return, and assumes this is one line of data. It then takes this line and will search for the first rule that shows the end of the first column.

The first column entry will then be properly *and independently* set with the correct offset, justification, font, and width. The process continues on a column by column basis until the end of the line. It then repeats for each needed line of the entire table.

As an optional "gee whiz" feature, a half vertical line space can also be provided by including an [esc]-h or \033h in any column of any line. This lets you attractively show your data.

If you are not using my gonzo justify routines, then replace {cl} {cc} {cr}, or {cf} with procs that do a suitable left, center, right, or fill justify. The *txtwide* command has to be replaced with whatever is setting your max column width. *changefont* also has to be replaced with whatever is setting your initial font selection for any column. Note that most any number of fonts can be mixed in any particular column you like.

Since *ctab* is in raw PostScript code, you might easily rearrange the scenery to suit yourself.

While *ctab* is intended for single line entries, any left or fill justified column entry is able to automatically break itself up into as many vertical lines as needed.

At present, you do have to bracket each entire data table with the *xpos* horizontal position, a *ypos* vertical position, an opening parenthesis, the data, a closing parenthesis, and a *ctab*. Which is no big deal. But, you can automate this by going to input file reading code, such as we saw back in August.

Note that some extensions on this new general *ctab* technique will let you custom fill in such things as tax or insurance forms.

Don Lancaster's

ASK THE GURU

February, 1989

The tearing method
SX cartridge refilling
Some thick paper ploys
PostScript font bitmaps
Colorease and Identicolor

For several years now, spunky and independent APDA was the place anybody could go to get Apple "insider" tech literatature. Besides stocking nearly all of the Apple publications (preliminary or otherwise) they also offered deep discounts on third party books and software.

Apple Computer has apparently taken over at least part of APDA, and is now once again directly distributing much of its own tech literature. While this should, in theory, improve service and speed up delivery, there are already rumors of sharply higher prices, particularly involving the IIgs (APW) and Mac (MPW) software development systems.

This new Apple service is called the *Developer Tools Direct* program.

The twin spectres of limited stocking of third party products and the limited availability to non-developers do seem to be rearing both of their ugly heads. Time will tell us whether this move is for better or for worse.

There is now an unbelievably vile atrocity included "free" with the new GS/OS operating system that renders it utterly and totally useless. You can not reset into an application, not even into APW. As a sadistic side effect, a reset can trash open files.

Such arrogance and stupidity has no place whatsoever in the Apple II universe. Whichever epsilon minus dreamed this one up should be staked to an anthill. Real soon like. When will they ever learn?

Do NOT under any circumstances use GS/OS for anything ever, until such time as an absolute and totally unconditional to-the-program reset is permanantly and acceptably restored.

Be sure to rush me any temporary fixes on this. The IIgs is now dead in the water, and likely to stay that way, unless something gets done fast.

Meanwhile, Apple does have one all around good deal involving the CD-ROM optical disks. They will now line you up with several third parties who will custom master your own personal optical CD-ROM disks for you, at an unbelievable price of only $15 each.

And there are only three rather small gotchas: You do have to buy 100 identical disks at once. You are limited to a paltry 80 megabytes max per disk. And this is a "one shot" promo deal aimed at getting you hooked on CD ROM.

Good old ProDOS AppleWriter is once again available through the *Sun Remarketing* folks at a cost of only $29. Several third parties are working on a IIgs upgrade. Stay tuned.

Tom Weishaar's great *Open Apple* magazine has changed its name to *A2 Central*. They also are now stocking the best of the hard-to-get Apple software and publications. His *Open Apple*, of course, has been far and away the very finest Apple publication anywhere ever.

Hewlett-Packard seems to be the first one to market with the newest *Canon RX* duplex laser printer. This jewel prints on both sides of the page and is only slightly slower than the usual SX engines as used in both the LaserJet II and LaserWriter II.

Duplex printing is essential for any book printing on demand and for lots of other production uses. It greatly simplified collating and dramatically reduces binding errors.

Unfortunately, that new LaserJet IID is totally brain dead and does not speak PostScript. In my opinion, this renders it utterly useless for any serious desktop publishing.

It would be reasonable to expect a duplex Apple LaserWriter using this engine. Hopefully coupled with those newer and faster versions of PostScript that have already obsoleted the NTX. A *Weitek* or other RISC engine would also be nice, as would refillable toner cartridges and removal of the insidious font lockout that is now present on *pathforall*.

There are also some third party schemes underfoot to dramatically improve halftone imaging on all the *Canon* laser engines. This is done by intensity modulating the laser dot size, rather than using the fixed dot size as is done in the stock Apple and H-P machnes. The *Visual Edge* enhancer by *Intel* is one example of this type of product. It does require some hardware printer mods.

Meanwhile, the PostScript juggernaut rolls on. *Tektronix* now has a new full color PostScript laser printer out that uses the *Sharp* thermal wax engine. And *Linotronic* has come out with a new el-cheapo *Linotron 200-P* PostScript phototypesetter. The max resolution is 1700 DPI, and the cost is down in the $29K range. This is far less than their older machines.

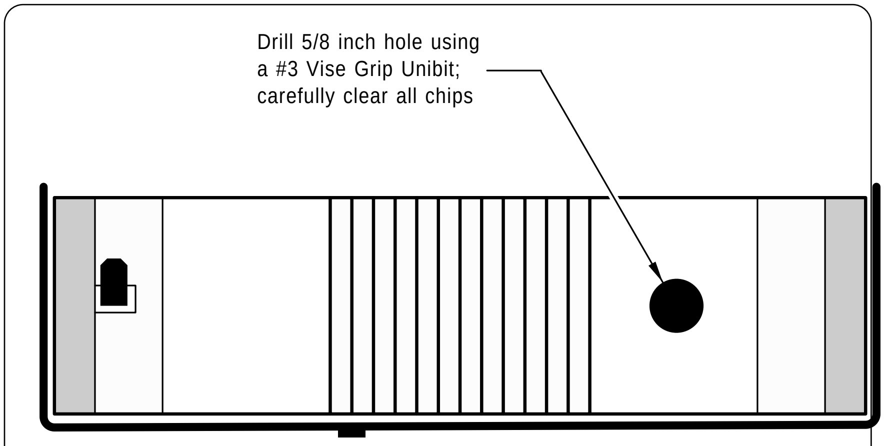


Fig. 1 – Adding a fresh toner filling hole to a SX cartridge.

Adobe Systems is finally releasing details on all their great new *Display Postscript* systems. They have both a free *Display PostScript Overview* for anyone and a $30 *Display PostScript Technical Reference* package for all of you serious developers.

Two of the active PostScript BBS systems now include (707) 882-2290 and (713) 688-1779.

Shoptalk. *Radio-Electronics* has bought the old *Popular Electronics* name from *Ziff-Davis* and now fully intends to aggressively revive both the intent and spirit of the original magazine. Carl and Jerry might even return. I suspect I might eventually end up involved with this.

The SPCA is after me over that scrawny and underfed wolf family that's trying to eek out a marginal subsistence on my doorstep. Those pups sure are looking rather pitiful. Scurvy and all.

So, now's a good time to remind you about our *Ask The Guru* volume I and II, along with the *Hardware Hacker II* books. For those of you into PostScript, we do have my *PostScript Show and Tell*, the *Intro to PostScript* video, and the *PostScript Work in Progress* disks. And we now do stock all of the usual *Apple* and *Adobe* books on PostScript.

At the very least, send along some Purina Wolf Chow and some Vitamin C. The real biggie this month does involve retreiving the actual bitmaps and/or scanmaps of any previously imaged PostScript fonts.

But first . . .

What is the "Tearing Method"?

It is an astonishingly fast and very simple way of taking apart and then analyzing virtually any machine language or assembly language code.

While full details do appear in my *Enhancing your Apple*, volume I, I'll give you a quick summary here.

The keys to the tearing method are to use page hilighters to color code a printed listing, and letting that listing self-reveal its structure, purpose, and function. Most of the method can be done by rote (I've even taught this to seventh graders), and the results will often happen ten times faster and ten times easier than any other method that I know about. A second secret is letting your subconcious work on the analysis for you, while you are doing simple and rote tasks.

The tearing method is also lots of fun. In fact, all of my microprocessor students freshly exposed to tearing last night refused to leave the lab, and I simply locked them in at 1 AM this morning.

Bet you can't tear just one.

Anyhoo, first be sure that you do totally and thoroughly understand exactly what the program does and how to use it. Then, you will load the program into a known location in your Apple. One simple way to do this is to fill the available RAM with "11" elevens and see exactly where the elevens are *not* after loading. My *absolute reset* or a similar utility may be needed for some protected code.

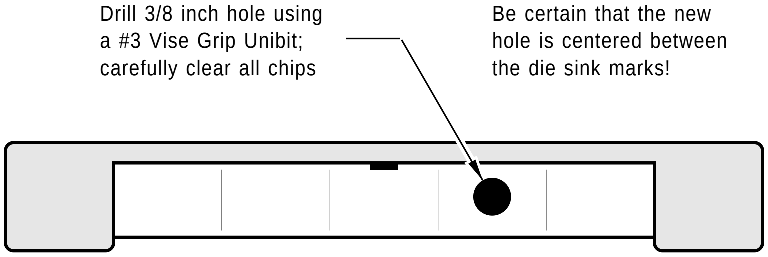

Drill 3/8 inch hole using a #3 Vise Grip Unibit; carefully clear all chips
Be certain that the new hole is centered between the die sink marks!

Fig. 2 – Adding a spent toner emptying hole to a SX cartridge.

You then create two disassembled listings and two hex dumps of your code, and then heavily reinforce them with Scotch tape or whatever.

Go through the listing and lightly and quickly scan it, seperating those main working code areas from the associated files. Important file clues are lots of question marks, plenty of BRK zeros, and oddball code used in strange ways.

Next, go through only the working code areas and paint each RTS green. The RTS command forms both the Achille's Heel and the Rosetta Stone to cracking anything. Then you paint these subroutine calls orange, noting that there are in-code calls, out-of-code calls, and finally and most importantly, calls to monitor locations and other fixed and known Apple resources. Create a cross reference list as you go along.

Your branches next go down in blue, using vertical and horizontal for upward going loops and diagonals for downward going breakouts and any bypasses. Your jumps and indirect jumps are then done in pink.

At this point, you temporarily set aside the working code and see what you can find out about the related files. First seperate the files by their vibes, seperating stuff that looks like it belongs together from any "right angle turns" in the bytes that seem to be popping up.

You then try to identify what a file is and what it contains. First and foremost are the ASCII text listings, which will have lots of "20" spaces with plenty of "4X" and "5X characters between them. You will note all similar patterns for lower case and high ASCII as well.

Other types of files that are easy to identify include the lists of addresses, shape tables, DOS and ProDOS command tables, and similar goodies.

You then go back into your most popular subroutines and see which of them are actually printing something to the screen or doing disk access.

Once you crack a sub, you will use your cross reference list and the *avalanche effect* to begin cracking all the routines that called this sub.

As you understand the purpose of each instruction, you make the usual loads and transfers green, the stack pushes or pulls yellow, and the stores pink. As each portion of the code is understood, a stripe is painted down the right side of the page, wide green for fully understood code and wide yellow for fully cracked files. This lets you keep score and mark your progress. Write down everything you learn next to each instruction.

The innermost secret to the tearing method is to avoid ever doing any thinking or any analysis until the last possible instant. Just keep coloring with the hilighters and they will do nearly all of the work for you.

Well, that is sort of a thumbnail sketch of what tearing is all about. And, yes, tearing will work on *any* make and model of microcomputer. Full details appear in *Enhance I*.

Have fun with this incredibly powerful analysis and study tool.

How do I Refill a Canon SX Toner Cartridge?

As we have seen in past columns, there is now as much as a 15:1 cost penalty in per-page toner costs when using those newer *Canon* SX laser printers over the older CX engines. On the older CX cartridges, you were able to buy cartridges for five bucks out of your Sunday paper and refill them up to seven times, bringing all your toner costs down into the 0.33 cents a page range that is cost competitive with jiffy offset printing.

Unfortunately, those SX cartridges do use a highly abrasive toner, combined with drums that are intolerably scratch sensitive.

So, while you can in fact reload SX cartridges, at present, you can not even remotely approach those CX cartridge economics. So, do consider this a progress report where I'll bring you up to date on what can and cannot be done at the present.

While it is difficult to even get a second SX reload, you can sometimes do so with the following tricks and techniques. Firstoff, you *immediately* remove the factory toner and give it to your friendly neighborhood diesel mechanic for use as valve grinding compound. Replace it with a good quality third-party refill toner.

Second, use a good quality drum lubricant, such as *Pixie Dust* or its equivalent. Do a very light dusting on every refill.

There are two refilling methods, the *Punch and Go* and the *Total Teardown*. I very much prefer punch and go, since this delivers far and away the lowest per-page toner cost to the end user. We charge $22 for local SX refills. Since this is a remote rural area, I can get away with such an outrageously high price. You can do the job yourself for as little as $7.50 and three minutes time.

The SX cartridge needs modified before you can refill it. Using a Vise Grip #3 *Unibit* and a very slow drilling speed, you drill the two holes shown in figures one and two. Drill upside down and be very careful to remove the single chip that the unibit provides. These two holes are then

```
%  vlinemap scans an imageproc by vertical pixel lines, and transforms it by mapproc.
%  Input pixelswide pixelshigh {imageproc}. For perspective, 3-D, and isometric cylinders.

/vlinemap {save /plrsnap exch def /imageproc exch def 300 mul 72 div cvi /pixelshigh exch def
300 mul 72 div cvi /pixelswide exch def 0 1 pixelswide {/slinenum exch def save /plrsnap1 exch
def gsave mapproc newpath slinenum 72 mul 300 div 0 moveto 0 pixelshigh rlineto 0 0.2 rlineto
0 pixelshigh neg rlineto closepath clip newpath imageproc grestore clear plrsnap1 restore} for
clear plrsnap restore }def

%  hlinemap scans an imageproc by horizontal pixel lines, and transforms it by mapproc.
%  Input pixelswide pixelshigh {imageproc}. For "star wars" lettering, etc.

/hlinemap {save /plrsnap exch def /imageproc exch def 300 mul 72 div cvi /pixelshigh exch def
300 mul 72 div cvi /pixelswide exch def 0 1 pixelshigh {/slinenum exch def save /plrsnap1 exch
def gsave mapproc newpath 0 slinenum 72 mul 300 div moveto pixelswide 0 rlineto 0.2 0 rlineto
pixelswide 0 neg rlineto closepath clip newpath imageproc grestore clear plrsnap1 restore} for
clear plrsnap restore }def

%  reportvlinemap compiles vlinemap into an array of form [ xpos0 ypos0 yheight0 ] . . .
%   [ xposN yposN yheightN ], and returns it to the host. This output may then be defined
%  as a single array, get run length encoded, or else converted to a bitmap.
%  To use, include reportvlinemap as the last command in your imageproc.

% WARNING: reportvlinemap is specific for version 38.0 and older LaserWriters.

/stall {50 {37 sin pop} repeat} def  % extra delay needed by some hosts.

/reportvlinemap {false charpath clip newpath -100 dup moveto 1000 0 rlineto 0 1000 rlineto
-1000 0 rlineto closepath clip clippath mark {{moveto}} {{lineto}} {{curveto}} {{closepath}}
pathforall ] dup /stripe exch def length 1 sub /stripelength exch def mark 0 13 stripelength
{/sposn exch def stripe sposn get stripe sposn 1 add get transform round cvi exch xshift sub
round cvi 75 add exch dup 3268 exch sub exch  stripe sposn 6 add get stripe sposn 7 add get
transform exch pop round cvi sub neg exch 1 index sub yshift sub exch} for] == flush stall} def

/xshift 0 def /yshift 0 def  % defaults

%  drawvlinemap converts an array of form [ X0 Y0 DY0  X1 Y1 DY1 . . . XN YN DYN ] into
%  the actual image, drawing one piece of one scan line at a time.

/drawvlinemap {/vlinearray exch def gsave 0 setlinewidth 0 setlinecap 72 300 div dup scale 0 3
vlinearray length 1 sub cvi {/vlp exch def vlinearray vlp get vlinearray vlp 1 add get moveto 0
vlinearray vlp 2 add get rlineto stroke} for grestore } def

%  drawhlinemap converts an array of form [ X0 Y0 DX0  X1 Y1 DX1 . . . XN YN DXN ] into
%  the actual image, drawing one piece of one scan line at a time.

/drawhlinemap {/hlinearray exch def gsave 0 setlinewidth 0 setlinecap 72 300 div dup scale 0 3
hlinearray length 1 sub cvi {/vlp exch def hlinearray vlp get hlinearray vlp 1 add get moveto
hlinearray vlp 2 add get 0 rlineto stroke} for grestore } def

%  createbitmap creates an "empty" bitmap of all zeros to be later filled from a vlinemap
%  or hlinemap array. Predefine pixellines and pixelsperline. Then call createbitmap.

/createbitmap { /pixelsperline exch def /pixellines exch def /bitmap pixellines pixelsperline mul 8
div cvi array def 0 1 bitmap length 1 sub {bitmap exch 0 cvi put} for} def

% setpixel sets a single bit in a bitmap. Enter with bitposition setpixel.

/setpixel {cvi dup 8 div floor cvi exch 8 cvi mod 1 index bitmap exch get exch 128 exch neg
bitshift or bitmap 3 1 roll put} def

% vlinetobitmap adds a vlinemap array of form [ X0 Y0 DY0 . . . X1 Y1 DY1 . . . XN YN DYN]
%  into an already exiting bitmap.

/vlinetobitmap {/blist exch def 0 1 blist length 3 div cvi 1 sub {/ptr exch def blist ptr 3 mul get
xshift sub pixelsperline mul blist ptr 3 mul 1 add get yshift sub add /strt exch def strt 1 blist ptr 3
mul 2 add get strt add 1 sub cvi {setpixel} for } for } def

%  bitmaptohost returns an existing bitmap to the host computer for capture and further use.

/bstall {60 {37 sin pop} repeat } def % lengthens host recording time

/bitmaptohost {bitmap{16(     )cvrs dup length 1 eq {(0) print flush} if print flush bstall} forall} def
```

Fig. 3 – PostScript scanmap and bitmap utilities.

capped with a nickel *Caplug* or else some very agressive tape.

There are three major steps to the refilling process. You first open the holding tank hole and carefully shake out the excess toner. Do this outside and avoid breathing any of the toner. You then reseal the holding tank hole, open the fresh toner tank hole, and pour in a bottle of refill toner.

Finally, you remove the old fusion wiper wand and peel and stick a new wiper pad in place.

Another tip: keep the green toner dial advanced all the way to nine for any and all rough drafts and for all internal use documents. Note that the *higher* the number, the *less* toner you will use. Cartridge life can easily be doubled with this simple technique.

I currently recommend using *Lazer Products* to supply toner, pixie dust, wiper pads, and drum recoating.

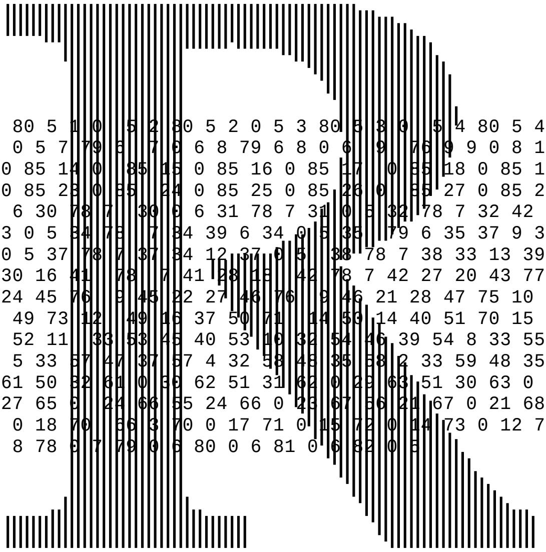
Using the utilities of figure three, here is how to convert a PostScript font Palatino "R" into a vertical scanmap . . .

```
/Palatino-Bold findfont [30 0 0 30 0 0] makefont setfont
/mapproc {100 100 translate 1 dup scale} def
/imageproc {newpath {2 2 moveto (R) reportvlinemap}} def
/xshift 431 def /yshift 425 def 30 20 imageproc vlinemap
```

. . . returns this scan list to the host . . .

```
[] [] [0 80 5 0 0 5 ] [1 80 5 1 0 5 ] . . . [81 0 6 ] [82 0 5 ] []
```

. . . which is easily edited into . . .



```
/Rscanmap [
0 80 5 0 0 5 1 80 5 1 0 5 2 80 5 2 0 5 3 80 5 3 0 5 4 80 5 4 0 5 5 80 5
5 0 5 6 79 6 6 0 5 7 79 6 7 0 6 8 79 6 8 0 6 9 76 9 9 0 8 10 0 85 11 0
85 12 0 85 13 0 85 14 0 85 15 0 85 16 0 85 17 0 85 18 0 85 19 0 85 20 0
85 21 0 85 22 0 85 23 0 85 24 0 85 25 0 85 26 0 85 27 0 85 28 78 7 28 0
8 29 78 7 29 0 6 30 78 7 30 0 6 31 78 7 31 0 5 32 78 7 32 42 3 32 0 5 33
78 7 33 41 4 33 0 5 34 78 7 34 39 6 34 0 5 35 79 6 35 37 9 35 0 5 36 78
7 36 36 10 36 0 5 37 78 7 37 34 12 37 0 5 38 78 7 38 33 13 39 78 7 39 31
15 40 78 7 40 30 16 41 78 7 41 28 18 42 78 7 42 27 20 43 77 8 43 25 23
44 77 8 44 24 24 45 76 9 45 22 27 46 76 9 46 21 28 47 75 10 47 19 31 48
74 11 48 17 34 49 73 12 49 16 37 50 71 14 50 14 40 51 70 15 51 13 43 52
65 20 52 45 16 52 11 33 53 45 40 53 10 32 54 46 39 54 8 33 55 46 38 55 7
32 56 47 37 56 5 33 57 47 37 57 4 32 58 48 35 58 2 33 59 48 35 59 1 32 60
49 34 60 0 32 61 50 32 61 0 30 62 51 31 62 0 29 63 51 30 63 0 27 64 52 28
64 0 26 65 53 27 65 0 24 66 55 24 66 0 23 67 56 21 67 0 21 68 58 18 68 0
20 69 61 13 69 0 18 70 66 3 70 0 17 71 0 15 72 0 14 73 0 12 74 0 11 75 0
10 76 0 9 77 0 8 78 0 7 79 0 6 80 0 6 81 0 6 82 0 5
] def
```

Here's one way to show a vertical scanmap . . .

```
100 200 translate 10 dup scale Rscanmap drawvlinemap
```

Both normal and greatly magnified results are shown above. Note that there is a 512 element array limit when using the stack to create the array. Use several arrays or else **put** to create arrays of 65530 elements or less. Scanmaps are usually more compact than bitmaps.

Fig. 4 – Converting a PostScript font to a scanmap.

How does the Thick Paper Ploy Work?

Way back in college, I tended to get very high grades on all of my lab reports. In fact, that's what got me started of into much of this current silliness. Since the statue of limitations is long since over, I will here reveal the secret to getting very high grades on lab reports. You simply hand in a report that is thicker than any of the others.

Now, there are all sorts of ways of creating a thick lab report, but far and away the most cost effective method is to use thick paper. I found this very high bulk and quite white paper called *biology filler* over in the bookstore. And that was the key secret to running away with all the marbles. Nobody ever caught on.

Seriously, the paper thickness can have a profound effect in the user acceptability of any desktop published product, especially if it is bound.

If a book or bound whatever looks a tad on the skimpy side, it will not "feel" right. Upgrading a weight or two on the paper might dramatically improve user acceptance.

On the other hand, if you do reach the thickness limits of your binding system, or if things get too fat, then using a lower bulk or a thinner paper will once again get you back into the acceptable range.

There are obvious limits of course. Too thick a paper and the costs go up and the feeding becomes a problem. Too thin a paper and you'll get print-through and an overall skimpiness.

Tell Me all about The Colorease Process

Colorease is a new way to convert any toner image into instant transfers of multiple color real ink that can end up on most any surface.

The process gets messy and labor intensive, but it can give you some outstanding results. It is not nearly as expensive as some competing color proofing schemes.

Here is how it works: You start with a special coated paper or else an instant transfer release liner. You then apply a thin layer of your first ink color, using special stainless steel grooved rollers. You harden the ink with a hair dryer, and then apply a photoemulsion overcoat. Then, you expose the ink through your laser created transparency, either with an u-v contact printer or by exposing it to the sun for a few minutes.

Developer is then wiped on, which removes the photo resist and all ink that is in unwanted places. You can repeat the process for as many colors as you like. Related materials are available for raised print thermography and for glass etching.

A more expensive and competing process is *IdentiColor*, while a much simpler and cheaper system is *Kwik Print*, available through the folks at *Light Impressions*.

What is This Month's PostScript Utility?

The one question most asked on our no-charge PostScript helpline is "How can I get a font bitmap into a storable or useable form?" This topic is less than adequately covered in the red, blue, and green books, yet the user demand is very high.

Normally, it is not a good idea to work with font scanmaps or font bitmaps in PostScript, since these both might violate device independence. Bitmaps will not scale upward without that "Hershey bar" effect setting in, and can also be extremely slow memory hogs. But, there are times and places when you might literally leap tall buildings in a single bound when you have scanmap or bitmap tools available.

For instance, I have long been interested in *non-linear* font transformations. All of the usual linear font matrices only let you shift, rotate, or scale *entire* font characters. I am far more interested in those non-linear transformations that reach in and individually change the position and size of each and every pixel scan line in the character.

Uses? Perspective lettering. True three dimensional lettering. Fonts forced to an exact outline. Salvadore Dali-like fonts that can lean or droop. Lettering on an isometric cylinder. Calligraphy on a parchment scroll. Twisted fonts. And, of course, nearly everyone wants those good old leaning "star wars" messages.

A ferinstance. As you know, I am very big on doing book-on-demand printing. In three illustrations in *Ask the Guru* volume II, there was some perspective lettering that used my old pixel line remapping techniques. The total imaging time for the lettering was over ninety minutes. This rather cramps your style when you intend printing the entire book one copy at a time in less than half an hour. Worse yet, that pixel remapping required many thousands of font creations that totally trashed over the font cache. This slowed down the next several pages, until the cache can recover.

By going to some post-processing scanmap and bitmap techniques, the imaging time was changed from over ninety minutes to under two seconds, a speedup of 2700:1, with a zero need for any actual font imaging.

At any rate, figure three shows you some PostScript utilities that get you from a string or other proc to either a scanmap or a bitmap.

I prefer the scanmaps. What these do is give you a list of where along each scan line to start and end an individual line of pixels. These can be related to a compression technique known as *run length encoding* that can often end up considerably more compact than "real" bitmaps.

There are two possible scanmap directions. A *vertical* scan map will give you vertical pixel bars. These are useful for perspective and 3-D lettering. A *horizontal* scanmap will give you horizontal pixel bars, handy for "star wars" images.

Figure three does show you several utilities that create the vertical and horizontal scan maps, image these scan maps, convert them over into bitmaps, and return the bitmaps or scanmaps to your host for further capture and processing.

There is one very big gotcha. The *reportvlinemap* routine will not work on certain newer PostScript printers. It is intended specifically for use with the "old" LaserWriter and LaserWriter Plus having version 38 or earlier software. A bug seems to have worked its way into the newer printer firmware that makes *reportvlinemap* rather tricky to use.

You can also use similar ideas on any system that has Display PostScript or on UNIX based PostScript printers that include the *frametofile*.

Note that the "old" LaserWriter is only needed for one post-processing pass. Once you have captured your scanmap or bitmap, any PostScript printer can be used for reprintings.

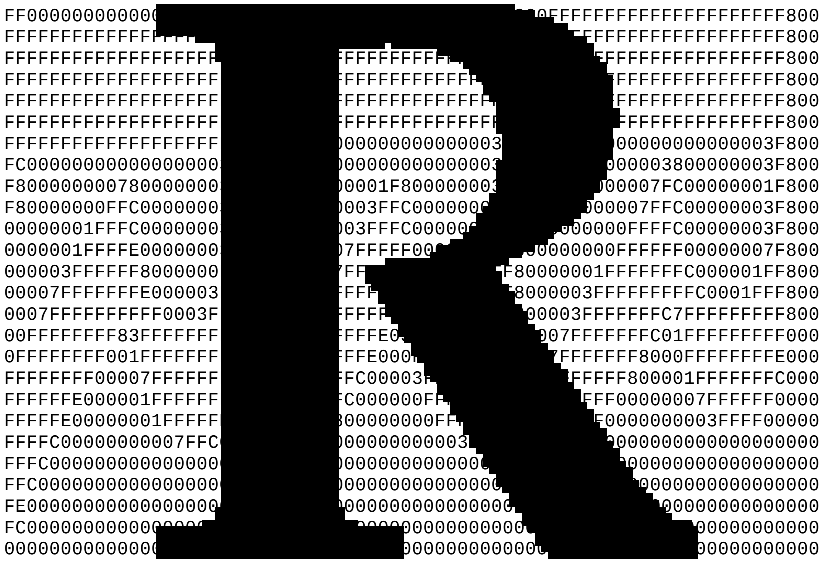
Figure four shows how to get a single PostScript letter into a scanmap, while figure five shows how to convert the same letter from a scanmap over to a bitmap.

Start with the **Rscanmap** array of figure four. The bitmap is then created this way . . .

96 96 createbitmap Rscanmap vlinetobitmap bitmaptohost

. . . Which returns this bitmap to your host . . .

```
/Rbitmap <
F8000000000000000000F800F8000000000000000000F800F8000000000000000000F800
F8000000000000000000F800F8000000000000000000F800F8000000000000000000F800
F80000000000000000001F800FC0000000000000000001F800FC0000000000000000001F800
```



```
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000
> def
```

Here’s one way to show the bitmap . . .

gsave 200 300 translate 96 72 mul 300 div dup scale 90
rotate true [96 0 0 -96 0 0] Rbitmap imagemask grestore

Both normal and greatly magnified results are shown above. The magnification is ratty looking since you are viewing all of the individual pixels magnified by a factor of 10X.

Fig. 5 – Converting a scanmap to a bitmap.

Don Lancaster's

ASK THE GURU

March, 1989

The $1300 LaserWriters
Secret non-putrid grays
Sources of ticket blanks
WPL font selection code
Gonzo justify templates

Just for kicks, try to calculate the effective baud rate of a single half-gigabyte CD-ROM disk that will get mailed once a month. Figure around ten of your baud "bits" per byte.

The answer is rather scary and has profound implications for the entire future of data communications.

Speaking of communications, be sure to get yourself a copy of *Signal*, from the *Whole Earth Review*. This $17 volume in the style and flavor of the original *Whole Earth Catalog* has a wide variety of comm resources in it, ranging the gamut from desktop publishing to interspecies communication. This one is in the "must have" category. Try your local bookstore.

The *Infocom* folks have lowered prices on many of their classic text-only adventures, down into the $14 range. They also have a free and full color *Escape* magazine that has now replaced their older *Status Line*.

Uh, I hate to ask this, but I can't get past zapping the green emerald in *Trinity*, and New Mexico is nowhere in sight. For a free *Incredible Secret Money Machine* book, how about you giving me a hint or two?

There's lots of exciting laser stuff happening. *Intel* has improved the process of modulating the size of laser printing dots, and their *visual edge* products can now dramatically improve the halftoning process. You can now do a 100 line halftone on a 300 DPI printer and still have 37 gray levels! This means that you can now attractively reproduce a real photo. And, yes it does work on the *Canon* SX engine. Three big limitations at present are that the *visual edge* is usually IBM pc based, that it isn't yet PostScript compatible, and that it is a voracious memory hog, needing as much as four additional megabytes.

Meanwhile, *Weitek* has a free new data booklet on their XL-8200 *Hyper-Script* processor. This beauty does some PostScript operations as much as 100 times faster than a LaserWriter Plus. On the other hand, it is only marginally faster than a LaserWriter

```
 pnd fontload.w
 ppr [L]
 ppr Font Picker:
 ppr ............
 ppr
 ppr Inserts font into current textfile position.
 ppr
 ppr13 Avant         0 Courier         29 Palatino
 ppr14 Avant b       1 Courier b       30 Palatino b
 ppr15 Avant i       2 Courier i       31 Palatino i
 ppr16 Avant bi      3 Courier bi      32 Palatino bi
 ppr
 ppr17 Bookman       8 Helvetica        4 Times
 ppr18 Bookman b     9 Helvetica b      5 Times b
 ppr19 Bookman i    10 Helvetica i      6 Times i
 ppr20 Bookman bi   11 Helvetica bi     7 Times bi
 ppr
 ppr25 Century      21 N Helvetica     12 Symbol
 ppr26 Century b    22 N Helvetica b   33 Zapf Italic
 ppr27 Century i    23 N Helvetica i   34 Zapf DIngbats
 ppr28 Century bi   24 N helvetica bi  99 Custom font
 ppr
 pinWhich Font? --> =$A
 pasX$A=$B
 pcs/$B/X/
 pqt
 pcs/$B/X /
 pqt
 p
 pasItalic=$B
 pasOblique=$C
 pas=$D
 pgo$A
13 pasBook$D=$D
! pasAvantGarde-$D=$D
 pgox
14 pasDemi$D=$D
 pgo!
15 pas$C=$D
 pgo13
16 pas$C=$D
 pgo14
29 pasRoman=$D
@ pasPalatino-$D=$D
 pgox
30 pasBold=$D
 pgo@
32 pasBold$B=$B
31 pas$B=$D
 pgo@
4 pasRoman=$D
# pasTimes-$D=$D
 pgox
5 pasBold=$D
 pgo#
7 pasBold$B=$B
6 pas$B=$D
 pgo#
1 pas-Bold=$D
0 pasCourier$D=$D
 pgox
2 pasBold$C=$C
1 pas-$C=$D
 pgo0
9 pas-Bold=$D
8 pasHelvetica$D=$D
 pgox
10 pasBold$C=$C

11 pas-$C=$D
 pgo8
22 pas-Bold=$D
21 pasHelvetica-Narrow$D=$D
 pgox
24 pasBold$C=$C
23 pas-$C=$D
 pgo21
25 pasRoman=$D
% pasNewCenturySchlbk-$D=$D
 pgox
26 pasBold=$D
 pgo%
28 pasBold$B=$B
27 pas$B=$D
 pgo%
17 pasLight=$D
^ pasBookman-$D=$D
 pgox
18 pasDemi=$D
 pgo^
20 pasDemi$B=$D
 pgo^
19 pasLight$B=$D
 pgo^
12 pasSymbol=$D
 pgox
34 pasZapfDingbats=$D
 pgox
33 pasZapfChancery-MediumItalic=$D
 pgox
99 pinCustom font name --> =$D
x p set d
 f//*&^%/
 y?
 p
 b
 f/*&^%//
 y?
 p
 pinHorizontal points -> =$A
 pinVertical points ---> =$B
 p
 pas/$D=$D
 f<<>>$D<
 y?
 p

 f<< findfont [$A 0 0 $B 0 0] makefont setfont><
 y?
 p
 pqt
```

Fig. 1 – WPL routine to insert PostScript fonts.

NTX with its "any day now" expected ROM and processor upgrades.

Over in the toner wars department, several firms are now offering the SX drum recoating at low cost. One good source is *Lazer Products*. What has *Canon* accomplished so far with their intentionally soft SX drums, all their "carbide tipped" toner, their grossly overpriced cartridges, and their "no reload" policy? Well, I feel they ...

(1) Have hacked off 90 percent of their existing users and just about 100 percent of their more knowledgeable perspective buyers;

(2) Have created the entire new toner refilling industry, where there are hundreds of sources of cartridge refilling services that in turn dramatically reduce the cartridges actually purchased from *Canon*;

(3) Have handed the entire low end desktop laser printer business over to NEC and their LC-890 with its much lower per-page toner costs; and, last but not least ...

(4) Have cut off their nose to spite their face.

What really galls me about all this are those *Canon* ads proudly proclaiming their super-hard longer life coating process they use on their high end copier drums.

I say again: Toner should cost the same as ink. 0.25 cents per page. No more. When it does, the true desktop revolution will at long last begin. Till then, you just ain't seen nuthin.

Alan Kalka has gotten his original PostScript BBS back on line, along with several brand new sponsors and lots of new equipment. Try dialing up (713) 688-1779. There is also a West Coast BBS at (707) 882-2390.

I finally found out what *MatroColor* is. It apparently is the leading color proofing and instant transfer process that will give you full color images that you can stick onto any surface. It is similar to the others we looked at in recent columns.

Let's see. I do have *Ask the Guru*, volumes I and II, and the *Hardware Hacker* volume II currently shipping, along with my *PostScript Show and Tell* for all major personal computers. Write or call per the *Need Help?* box if you need more info.

Our biggie this month is a three column gonzo layout template having auto left/right and built-in figures.

But first, these topics . . .

What Good is WPL?

WPL is that unusual supervisory *Word Processing Language* built into *AppleWriter*. With fresh new copies of ProDos Applewriter 2.1 and both manuals now once again available for only $29 from *Sun Remarketing*, now might be a good time to review *WPL*.

What WPL does is automate most any list of word processing tasks. It is sort of similar to an EXEC file or some type of "C" shell supervisor that you would find in the other personal computers. As a rudimentary language, it does support labels, strings, comments, numeric variables, math operations, subroutines, loops, gotos, and simple testing decisions.

With WPL, you can easily create help screens, produce form letters, get user inputs, do auto-formatting, do any automated search-and-replace

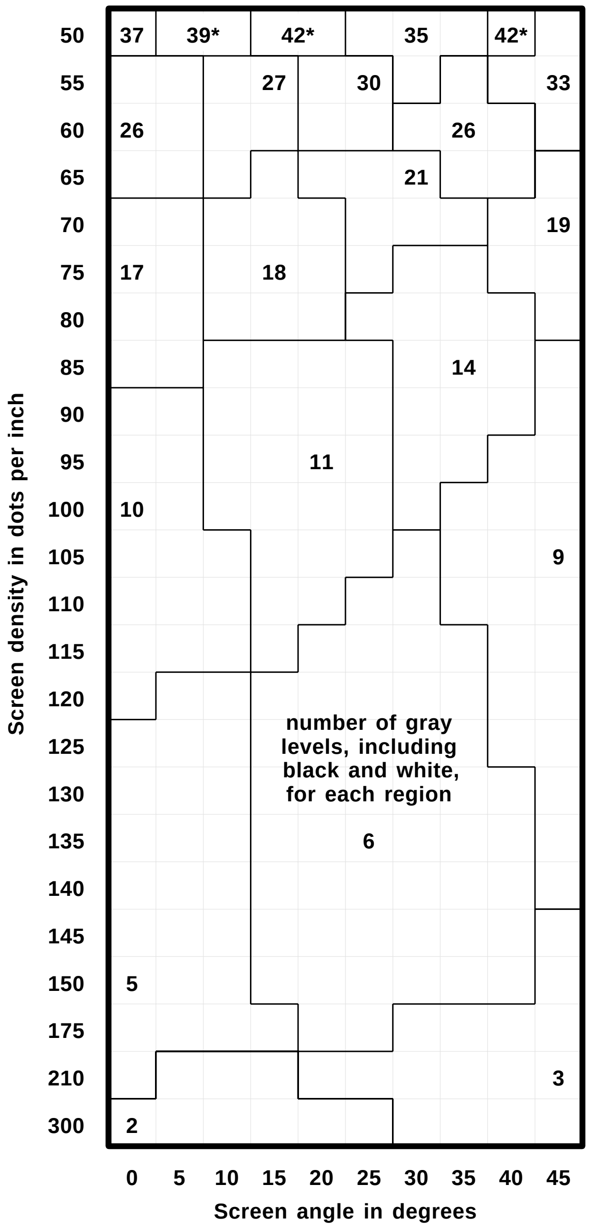


Fig. 2 – The top secret map to those hidden LaserWriter grays.

operations, or do any of dozens of other unique "higher level" tasks that involve word processing. It is mainly because of WPL that I still make exclusive use of *AppleWriter* for *all* of my desktop publishing and *all* of my book-on-demand printing. I have yet to find anything anywhere else that even comes remotely close.

A WPL instruction file is up to 2K characters long, and can be made longer using the *WPL Expansion Kit*. To run a WPL program, you simply do a [P] DO MYCOMMANDS.WPL, or whatever. The *WPL* program gets interpreted a line at a time, and then carries out various word processing tasks for you.

Each instruction line starts either with a label or a space. Your next character is interpreted as a control character. For instance, to go to the beginning of your file, your WPL line would be *(space) b*. The lines will get done in order, unless you have a repeat loop, or unless a zero value or an "unfound" search bypasses a command line. It is amazing how much WPL does with how little.

One example WPL program would do book on demand printing, by first clearing and loading the page files needed to print all of the rear page sides in reverse order. Then it will ring a bell, let you reload the pages, and then print all of the front page sides in reverse order, thus delivering to you a ready-to-bind book.

As a different example, figure one shows you a new WPL program that solves a sticky problem of mine. Any time you are writing your own PostScript code, you have to enter a long definition to create a font. Something like */Bookman-DemiItalic findfont [9 0 0 9.25 0 0] makefont setfont*, or whatever. It is super important to both remember and get all of those wierd font name spellings correct.

On an *[open-apple]-f*, command this WPL program creates a screen that will give you a list of all those available fonts. It then asks you to make a font selection, followed by a font width and font height in points. Finally, it will insert that correctly spelled definition into your current text file, and automatically return you to your word processing.

Around seven easy and intuitive keystrokes can replace several dozen very hard-to-remember ones.

You will have to predefine your *[open-apple]-f* as the glossary entry that executes this WPL routine.

WPL was Paul Lutus at his very best. The astounding thing is that the code for the entire WPL language is only several hundred of the tightest written bytes you have ever seen.

For additional WPL use ideas, see the *WPL Expansion Kit* from *Thompson and Thompson*, or the various WPL programs in my *AppleWriter Cookbook*. The latter also contains a complete disassembly script that does reveal the innermost WPL workings.

Where are all Those $1200 PostScript LaserWriters?

All over the place actually. But it does take some patience and some risk to put one together. It also does take an awful lot of being in the right place at the right time.

Here’s how this scam works: The engines used on an Apple LaserWriter Plus and a Hewlett-Packard LaserJet I are nearly identical, and you can simply plug a LaserWriter Plus board directly into a LasserJet I *Canon* CX engine. Some additional I/O circuitry consisting of a switch and a connector or two is quite easy to throw together out of *Radio-Shack* component parts.

There are just enough trade-in and repair LaserWriter Plus boards "out there" to make this a viable route to a low cost PostScript printer. The current street price is typically $450 for a trade-in board.

There is a glut on the market for used LaserJet I’s, because they are both brain dead and incapable of even speaking PostScript without a very costly add-on. The current street price is around $700.

There’s also an untrue myth that implies that older CX engines aren’t as black as the newer SX engines. In fact, the latest of the third-party toner refills for those CX machines are just about as black, and they can give you per-page toner costs as low as *one-fifteenth* that of the current SX costs. Thus, the older machines are actually *better* suited to serious production work. And, as far as service life goes, you can keep bolting *H-P* parts onto these forever. One of mine is rapidly approaching 400,000 copies.

There are three versions of LaserWriter boards available. These now include versions 23.0, 38.1, and 43.2, the latter often erronously referred to as a "version 47" ROM.

```
gonzo begin /workdict 100 dict def workdict begin

/ytop 550 def
/ypos ytop def
/ybot 50 def
/xpos 125 def
/txtwide 450 def

/pagenum 1 def
/pnxpos txtwide 2 div xpos add def
/pnypos ytop 19 add def

/showpagenumber {/pagenum pagenum 1 add def  pnxpos pnypos pagenum
(      ) cvs cc} def

/colcheck { ypos ybot le {showpage showpagenumber /ypos ytop def} if} def

/font1 {/Bookman-Light findfont [9 0 0 9 0 0] makefont setfont} def
/font2 {/Bookman-DemiItalic findfont [9 0 0 9 0 0] makefont setfont} def
/font3 {/Bookman-Demi findfont [9 0 0 9 0 0] makefont setfont} def

startgonzo
Your textfile starts here . . . .
                              . . . .  continues \0333here\0331 . . .
                                                              . . . . and ends here.
\033x

showpage
```

Fig. 3 – A simple "just dump the textfile" guru template.

You can find the version by giving a *version == flush* command, or else by looking at the test page. A (3.0) is version "47", while the lower numbers will be the older versions.

For most people most of the time, the latest version "47" ROM's are far and away the best choice, since this is the fastest code having the fewest major bugs in it. On the other hand, the version 38.1 chips let you bypass the font lockout on *pathforall*. This is an incredibly powerful tool that is conspicuously absent from the newer ROM chip sets.

For any given version, you can either obtain "one-half" of your ROM set, which gives you a 11 font LaserWriter, or else the "entire" ROM set, which instead will give you a 35 font *LaserWriter Plus*.

One very important gotcha: The keying on their main connectors is different, and the Apple board plugs in "upside down" compared to the HP board. Make sure you do thoroughly understand what you are doing before you make your final connections.

Here are some resources to get you started on all this: Used LaserJets are available from your Sunday paper or any of dozens of *Computer Shopper* advertisers, as are several brand new CX engines of dubious origin. Two sources of the LaserWriter boards are *Pre-Owned Electronics* and Richard Harold at *Shreve Systems*.

A complete set of quite simple conversion plans is available for $10 from *Custom Technology*, while the thorough repair and maintenence manuals are available directly from *Hewlett-Packard*. Their manual part number for the CX engine is #02686-90904, while the newer SX manual is numbered #33440-90904.

Already converted, pretested, and ready-to-run units are available from either *Thompson and Thompson* or from *Custom Technology*, while the toner refilling materials and supply kits are now available through Arlin Shepard at *Lazer Products*.

Tell me Once Again About Those Non-Putrid Grays

I sure have been getting a lot of calls on this lately, so let's us do a spring rerun here. The overwhelming majority of LaserWriter applications software and most users all seem to

```
%  Copyright c 1989 by Don Lancaster and Synergetics, 746 First Street, Thatcher,
%  AZ, 85552 (602) 428-4073. All rights reserved. Personal use permitted so long
%  as this header remains present and intact. Work in Progress disk costs $39.50.

%  This template intended for use with gonzo.dl.6 and ps.util.1 or later.

/persistent true def  persistent {serverdict begin 0 exitserver } if

/hack.II.temp.1 1000 dict def hack.II.temp.1 begin

/font0 {/Times-Bold findfont [54 0 0 54 0 -32] makefont setfont} def
/font1 {/Times-Roman findfont [9.75 0 0 9.75 0 0] makefont setfont} def
/font2 {/Times-Italic findfont [9.75 0 0 9.75 0 0] makefont setfont} def
/font3 {/Times-Bold findfont [9.75 0 0 9.75 0 -6] makefont setfont} def
/font4 {/Times-Roman findfont [9 0 0 9 0 0] makefont setfont} def
/font5 {/Times-Italic findfont [9 0 0 9 0 0] makefont setfont} def
/font6 {/Times-Bold findfont [9 0 0 9 0 0] makefont setfont} def
/font7 {/Helvetica findfont [9 0 0 9 0 0] makefont setfont} def
/font8 {/Helvetica findfont [8 0 0 8 0 0] makefont setfont} def
/font9 {/Helvetica-Bold findfont [9 0 0 9 0 4] makefont setfont} def
/font- {/ZapfDingbats findfont [9 0 0 9 0 4] makefont setfont} def
/font= {/Symbol findfont [9 0 0 9 0 0] makefont setfont} def

/txtwide 155 def /yinc 10.5 def /charstretch  0.2 def /spacestretch 0 def /maxsstretch 2.5 def
/maxcstretch 1 def /dropcount 3 def /dropindent 38 def /lastlinestretch 0.4 def /pm 0 def /ytop
720 200 sub def /ypos ytop def /ybot 80 def /yparendadj 0 def /pmnorm 10 def % paraindent

/amacro {(zy0)  stringmacro} def   % start drop cap
/bmacro {(iFy1) stringmacro} def   % finish drop cap
/cmacro {(zyC3) stringmacro} def  % centered title
/dmacro {(pF1)  stringmacro} def   % normal text

/rightpage true def /leftmargin {rightpage {60}{60} ifelse} def /xpos leftmargin def /colspace 170
def /colbot 60  def /coltop {700 colbot sub yinc div floor yinc mul 0.1 add colbot add}def
/totalcolumns 3 def /#column 1 def

/collimits [coltop colbot coltop colbot coltop colbot] def /starttext {/#column 1 def /ytop {collimits
0 get} def /ybot {collimits 1 get} def /ypos {ytop} def /xpos {leftmargin} def /justx (justL) def
totalcolumns {colcheck} repeat /firstuseflag true def} def

/header {titlepage {gsave leftmargin coltop 50 sub translate /colcheck {} def /txtwide 1000 def
/indent 4 def indent 33 (\0330Hardware Hacker\0331) cl indent 7 date cl indent 62
(\0332\0331Don Lancaster's\0332) cl 6 setlinewidth 1 setlinecap 3 23 moveto widthhold 6 sub 0
rlineto gsave stroke grestore nuisance begin 4.5 setlinewidth bestgray lightgray stroke black
end /boxwide 160 def widthhold boxwide sub 15 sub boxwide 80 80 4 1.5 grabbox  /bb {bt bh
sub} def boxpath gsave 1 setgray fill grestore gsave 4 setlinewidth stroke grestore gsave 2.5
setlinewidth nuisance begin lightgray end stroke grestore newpath gsave bl 23 moveto 0.99
setgray 4.5 setlinewidth 0 setlinecap -5 0 rlineto stroke br 23 moveto 0.99 setgray 4.5
setlinewidth 0 setlinecap 5 0 rlineto stroke grestore /yinc 12 def bw 2 div bl add 61 blurb
keystoneshow {cck}{cc} ifelse grestore} {leftmargin coltop 15 add moveto widthhold 0 rlineto 1
setlinewidth stroke leftmargin coltop 15 add 16 add moveto widthhold 0 rlineto 0.5 setlinewidth
stroke rightpage {0 0 (\0332\0331) cc leftmargin widthhold add coltop 15 add 4 add date
cr}{leftmargin coltop 15 add 4 add (\0332\0331Hardware  Hacker) cl} ifelse} ifelse} def

/footer {leftmargin colbot 12 sub moveto widthhold 0  rlineto 0.5 setlinewidth stroke gsave
leftmargin  rightpage {widthhold 35 sub add} {35 add} ifelse colbot 12 sub translate newpath -17
34 7 14 3 0.5 grabbox /bb {bt bh sub} def boxpath gsave 1 setgray fill grestore stroke   0 0
(\0332\0331 ) cc    0 -3.5 pagenum cc grestore} def

/boilerplate {/widthhold colspace 2 mul txtwide add def save /bsnap exch def /font0
{/Palatino-Bold findfont [30 0 0 30 0 0] makefont setfont} def /font1 {/Palatino-Bold findfont [12 0
0 12 0 0] makefont setfont} def /font2 {/Palatino-Bold findfont [10 0 0 10 0 0] makefont setfont}
def header footer clear bsnap restore titlepage {/tadj {50 yinc div floor yinc mul yinc 2 mul add}
def} {/tadj 0 def} ifelse collimits dup dup dup dup dup 0 get tadj sub 0 exch put 2 get tadj sub 2
exch put 4 get tadj sub 4 exch put} def

/startpage {boilerplate /#column 1 def /ybot -99999 def} def

/colcheck {ypos ybot le {/#column #column 1 add def totalcolumns #column lt {showpage quit} if
{/ytop {collimits #column 1 sub 2 mul get} def /ybot {collimits #column 1 sub 2 mul 1 add get}
def /xpos xpos colspace add def ytop ybot sub yinc gt {exit} if /#column #column 1 add def
totalcolumns #column lt {showpage quit} if} loop /ypos ytop def} if} def /fullheight {coltop colbot
sub yinc add} def

/keystonewide 300 def /keystonedelta 5 def /keystoneshow false def /gotarline {save /ksnap
exch def /justifylastparline true def /txtwide keystonewide def xpos exch ypos exch cf clear
ksnap restore /keystonewide keystonewide keystonedelta add def /xpos xpos keystonedelta
sub def /ypos ypos yinc sub def} def
/crk { /msg exch def /ypos exch def keystonewide sub /xpos exch def {msg (
) search {exch pop exch /msg exch def gotarline }{gotarline exit} ifelse} loop} def /gotacline
{save /ksnap exch def /justifylastparline true def /txtwide keystonewide def xpos exch ypos
exch cf clear ksnap restore /keystonewide keystonewide keystonedelta add def /xpos xpos
keystonedelta 2 div sub def /ypos ypos yinc sub def} def /cck { /msg exch def /ypos exch def
keystonewide 2 div sub /xpos exch def {msg (
) search {exch pop exch /msg exch def gotacline } {gotacline exit} ifelse} loop} def
```

Fig. 4a – A fancy three column gonzo auto-layout template . . .

have their collective hearts set on using the seventeenth lousiest gray that is available from their machines.

Now, that may be their trip and more power to them, but all this nonsense is giving 300 DPI printing a bad name. In reality, any LaserWriter can give you grays so fine that they look like an india ink wash.

Figure two once again shows you the LaserWriter secret gray map that reveals to you all of the hidden and all of the vastly better grays.

Any gray will trade off the number of gray levels shown inside the map against the coarseness or the *screen density*. Screens are only provided for certain angles of tilt, as shown along the bottom of the map. Since integer arithmetic gets involved and since everything has to "tile" itself correctly, these are the only available grays. Should you request a different screen density or angle, you'll always get one of the default ones shown in the map instead.

```
/denselisting{/bc br bl sub 2 div bl add def /txtwide bw 10 sub def /ybot -99999 def
/font0{/Helvetica-Bold findfont [10 0 0 10 0 0] makefont setfont} def
/font1{/Helvetica findfont [7 0 0 7 0 0] makefont setfont} def
/font2{/Helvetica-Oblique findfont [7 0 0 7 0 0] makefont setfont} def
/font3{/Helvetica-Bold findfont [7 0 0 7 0 0] makefont setfont} def
/font4{/Courier findfont [7 0 0 7 0 0] makefont setfont} def
/ypos bt 10 sub def /xpos bl 6 add def /yinc 8 def /amacro {(hy) stringmacro}
def /emacro {/yparendadj yinc 2 div def} def} def

/normallisting {/bc br bl sub 2 div bl add def /txtwide bw 10 sub def /ybot -99999 def
/font0{/Helvetica-Bold findfont [12 0 0 12 0 0] makefont setfont} def
/font1{/Helvetica findfont [8.5 0 0 8.5 0 0] makefont setfont} def
/font2{/Helvetica-Oblique findfont [8.5 0 0 8.5 0 0] makefont setfont} def
/font3{/Helvetica-Bold findfont [8.5 0 0 8.5 0 0] makefont setfont} def
/font4{/Courier findfont [8.5 0 0 8.5 0 0] makefont setfont} def
/ypos bt 20 sub def /xpos bl 20 add def /yinc 10 def /pmnorm 60 def
/amacro {(Lhy) stringmacro} def /bmacro {(1py) stringmacro} def
/cmacro {(3z) stringmacro} def} def

/showfigborder true def /showfigtitle true def /yfigadj 0 def

/figparams {/blw 0.5 def /brad 0.01 def /bh {figheight  yfigadj cvi yinc cvi mod add yinc div floor
yinc mul} def {/bt bb bh add yinc 0.2 mul sub def } {/bb bt bh sub 5 sub def} ifelse boxpath
showfigborder {stroke} if showfigtitle { save /tsnap4 exch def /txtwide bw def 55 changefont bl
bb yinc 1.25 mul  sub title cf clear tsnap4 restore} if } def /uppershift  {bh yfigadj add yinc div
ceiling yinc mul sub yinc sub} def

/figure1ll {/bl {leftmargin 1 add} def /bw {txtwide} def /bb collimits 1 get yfigadj add def true
figparams collimits dup 1 get bh add yfigadj add 1 exch put /yfigadj 0 def} def

/figure1lc {/bl {leftmargin txtwide add 1 add} def /bb collimits 3 get yfigadj add def /bw {txtwide}
def true figparams collimits dup 3 get bh add yfigadj add 3 exch put /yfigadj 0 def} def

/figure1lr {/bl {leftmargin colspace 2 mul add 1 add} def /bb collimits 5 get yfigadj add def /bw
{txtwide} def true figparamscollimits dup 5 get bh add yfigadj add 5 exch put /yfigadj 0 def} def

/figure1ul {/bl {leftmargin 0 add 1 add} def/bt collimits 0 get 6 add def /bw {txtwide} def false
figparams collimits dup 0 get uppershift 0 exch put /yfigadj 0 def} def

/figure1ur {/bl {leftmargin colspace 2 mul add 1 add} def /bt collimits 0 get 6 add def /bw
{txtwide} def false figparams collimits dup 4 get uppershift 4 exch put /yfigadj 0 def} def

/figure2lr {/bl {leftmargin colspace add 1 add} def /bb collimits 3 get yfigadj add def /bw
{colspace txtwide add} def true figparams collimits dup 3 get bh add yfigadj add 3 exch put
collimits dup 5 get bh add yfigadj add 5 exch put /yfigadj 0 def} def

/figure2ul {/bl {leftmargin 0 add 1 add} def/bt collimits 0 get 6 add def /bw {colspace txtwide
add} def false figparams collimits dup 0 get uppershift 0 exch put collimits dup 2 get uppershift
2 exch put /yfigadj 0 def} def

/figure2ur {/bl {leftmargin colspace add 1 add} def /bt collimits 0 get 6 add def /bw {colspace
txtwide add} def false figparams collimits dup 2 get uppershift 2 exch put collimits dup 4 get
uppershift 4 exch put /yfigadj 0 def} def

/figure2ll {/bl {leftmargin 0 add 1 add} def /bb collimits 1 get yfigadj add def /bw {colspace
txtwide add} def true figparams collimits dup 1 get bh add yfigadj add 1 exch put collimits dup 3
get bh add yfigadj add 3 exch put /yfigadj 0 def} def

/figure3ll {/bl {leftmargin 0 add 1 add} def /bb collimits 1 get yfigadj add def /bw {colspace 2 mul
txtwide add} def true figparams collimits dup 1 get bh add yfigadj add 1 exch put collimits dup 3
get bh add yfigadj add 3 exch put collimits dup 5 get bh add yfigadj add 4 exch put /yfigadj 0
def} def

/figure3ul {/bl {leftmargin 0 add 1 add} def /bt collimits 0 get 6 add def
/bw {colspace 2 mul txtwide add} def false figparams collimits dup 0 get bh sub yfigadj sub yinc
sub 0 exch put collimits dup 2 get bh sub yfigadj sub yinc sub 2 exch put collimits dup 4 get bh
sub yfigadj sub yinc sub 4 exch put /yfigadj 0 def} def

end
```

Fig. 4b – Three column gonzo template, concluded . . .

Your command for the overall best gray is *106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen*.

For the densest useful gray, try a *135 25* before your *setscreen* proc. This does take careful paper selection and attention to detail.

For a useful "repro" gray that will get reduced somewhat before you do your final printing, try a *85 35* before your *setscreen* proc.

The stock putrid gray #17 can be restored with a *53 45* prefix. To get the utmost prepress quality off of any gray image, first print up two totally black pages.

The complete details on where and why all these numbers come from did appear in the March 1987 issue of *Computer Shopper*, or else are found in my *Ask the Guru*, volume I.

Where is the Winning Ticket?

Right inside your PostScript laser printer, of course. Those custom and sequentially numbered tickets for raffles, plays, civic events, or whatever can be a very hot seller. These are all quickly done on your LaserWriter.

We saw my sequential numbering step-and-repeat routines back in the July 87 issue of *Computer Shopper*, also available in volume one of *Ask The Guru*.

One reader suggested perforating the tear-off tab of your own tickets by using a threadless and bobbinless sewing machine. Note that this is an extremely dangerous process, owing to the enormous hostility it evokes in the female of the species.

Otherwise, you can get various sized ticket blanks through your local *Paper Plus* store, or from any of a number of wholesale suppliers. For instance, *Blanks USA* will send you 300 free and prenumbered large sample tickets. Two of their competitors include *Ticket Express* and the folks over at *Quick Tickets*.

Please let me know all your laser ticket printing experiences, so I can pass them on to the others.

What is This Month's PostScript Utility?

Back in October of 1988, I showed you my gonzo justify routines that could give you exceptional quality typesetting, especially at 300 DPI. Some readers have asked for more detailed examples on how you gonzo justify real world output. A key here is to use one or more *templates*, and we will shortly look at both a very simple and a very detailed one.

But first, let us review what the advantages and disadvantages of the "gonzo method" are. The disadvantages are simple:

(1) You have to think while you are using them and need at least a smattering of PostScript knowhow or programming ability, and ...

(2) You will not get the final page image on your screen, unless you are already using Display PostScript.

Here are the main advantages:

(1) Its cheap, because all you need is your favorite word processor;

(2) You have device independence, since any computer at all works fine. The author, editor, printer, and proofreader, could all be using wildly different brands of computers and word processors anywhere in the world on virtually any network;

(3) Your files are very compact, typically 10K for a three column page layout with two figures;

(4) Production print speeds can be insanely faster, typically two extra seconds per three column, dual figure page after a simple pseudocompile is completed;

(5) Text, artwork, and graphics can all be done integrated within one single word processing program;

(6) Justification quality at 300 DPI is exceptionally high, including the automatic drop caps, hanging punctuation, a five stage progressive fill justification, the full kerning and tabbing, largely unlimited font choices including fractional sizes; and ...

(7) You can drop down into "raw" PostScript, at any place, any time, for any reason, rearranging all of your scenery to suit yourself. There are no "WhaddayameanIcant's" ever.

At any rate, a *template* is the key to using any page layout application, gonzo or otherwise. A template is simply a set of rules that set the style and the "vibes" of the document you are going to produce. The important values in a template will include the margins, the number of columns and their widths, all your font selections, handling of left versus right pages,

```
gonzo begin hack.II.temp.1 begin printerror /rightpage true def /titlepage true def /date (March,
1989) def /pagenum (15.1) def /posfig1 {figure2lr} def /showthegrid false def /dropindent 31 def
/keystonewide 103 def /keystonedelta 8 def /keystoneshow true def

/blurb
(A new "disco" circuit
Zero crossing detection
AC power load interface
Phase controlled dimming
Dialog information services
) def

% ///// BORDERS AND TITLES ////

startpage

% ///// INCLUDED FIGURE STARTS HERE /////

/figheight 200 def /yfigadj 4.5 yinc mul def

/title (\0337Fig. 1 – This ac power output interface will let you directly control 100 watt lamps
and other high power loads from your microcontroller or your personal computer. The special
phototriac optocoupler provides safety isolation. Note that a low logic input will light the lamp.)
def

posfig1 save /figsnap exch def

bl 19 add bb -8 add 11 setgrid clear line1 13 10 mt 2 u 2 l 2 d 15 14 mt dot 15 14 mt 6 l 4 d 11.5
14 hresistor 9 6 mt 1 d 6 l 6 5 hresistor 10 6 mt 3 d 7 l 4.5 5 mt rarrow 4.5 3 mt rarrow 24 14 mt
gsave 1.5 dup scale rarrow grestore 24 12 mt gsave 1.5 dup scale rarrow grestore 9 6 mt 6 (
)(+OUT NC -OUT)(+IN -IN NC) dipdraw line2 23 14 mt 8 l 6 d 6 r 4 u 2 r 18 14 mt incadlamp 15
11 mt triac

font1 18 16.3 (\0331100 watt) cc 18 15.5  (\0331light bulb) cc 24 12.8 (110 VAC) cc 17.5 9.8
(2N6154) cc 17.5 9 (TRIAC) cc 11.6 14.7 (220\0333W\0331) cc 6.1 5.7 (330\0333W\0331) cc 5
9 (MOC 3010)cc 5 8.2 (phototriac) cc 5
7.4 (optocoupler) cc 2.7 4.7 (+5V) cr 2.7 2.7 (PORT) cr

font2 /yinc 0.8 def 15.5 5.0
(\0332WARNING:
Extreme shock hazard on
right half of this circuit!) cl

clear figsnap restore

% ///// INCLUDED FIGURE ENDS HERE ////

% ///// ACTUAL TEXT STARTS HERE //////

starttext
startgonzo
\033a
S
\033b
everal helpline callers have now asked me what the main differences are between that
\0334BSEET\0331 supertech degree and the \0334BSEE\0331 engineering degrees. The quick
answer is "around one million dollars or so".\033p
That is roughly how much extra lifetime income the \0334BSEE\0331 degree will garner on the
average, including the benefits, perks, retirement plans, any investments, the other amenities,
and also allowing for inflation.\033p
It is no secret that the technicians and supertechs will often do all the work and the engineers
get all of the credit, all of the pay, and all of the promotions. Not to mention both an office and a
real desk.
Many of the larger and "old line" electronics outfits tend to treat their techs and supertechs as
second class citizens, severely limiting all of their advancement and salary opportun- ities.
These problems are especially acute in aerospace and defense.
So, \0332if\0331 you can handle all the need- ed math and can pass all the required
non-engineering courses, then that \0334BSEE\0331 will offer something around a
\033426.5\0331 decibel better cost/ benefit ratio over the \0334BSEET\0331 degree.
The helpline response over that fluxgate magnetometer compass we looked at back in the
December \0334RE\0331 has been utterly astounding, and I do thank you. Another source for
the fluxgate magnetometer compass kits is \0332Rusty Circuits\0331.
\0332Radio Shack\0331 also now has a low cost solid state fluxgate automotive compass. This
does appear to be a two-piece unit that has the fluxgate sensor windshield mounted by way of
a short length of five conductor cable. The display itself is a servo- like pair of coils that can
activate a magnetized compass rose disk. The accuracy does seem very limited, but it costs
only \0334$49.95\0331. This should hack beautifully. More on this whenever.
Radio Shack also has a new and "intelligent" power strip that turns on all of your computer
peripherals or home video accessories whenever a main load is switched on or off. A few
helpline callers have been asking for circuits to do this.
Several of you Canadian readers have been wondering why very few of those smaller
electronics outfits will even give them the time of day. The response is that there are more than
enough hassles involved that it is almost always a net loss of energy and time and money to do
so.
```

Fig. 5a – A sample three column gonzo textfile . . .

all the rules for inserting figures or artwork, and any command macros that can make life easier.

You can either persistently download all of your templates whenever printer power is first applied, or else tow them along as a prolog to each and every file. If you go the persistent download route, your files will be shorter, will download faster, and a person much less knowledgeable in PostScript can use them in an office environment. On the other hand, the prolog templates at the start of each document are easier to change and customize on the fly.

Let's look at two new examples of templates. One is quite simple, and the other rather complex.

Figure three will show you a "just dump the text" template that simply sets margins and fonts, and gives you a centered top page number on all but the first page. By adding a suitable letterhead procedure plus a curve-traced signature, you can convert this into any custom business letter template of your choosing.

To use this template, set it up as a prolog to your text. Then insert the *gonzo begin startgonzo* command, in turn followed by your text. The text can be from most any program on any computer. To break out of your gonzojustify, use an *[escape]-X* or an *\033X*, followed by a *showpage*.

Figure four does show you the template that I am now using for all the book on demand published *Hardware Hacker* reprints, volume II.

This gives you a three column, fill justified page layout that includes seperate treatments of title pages, left pages, and the right pages. That text automatically "flows" around figures that can be inserted in any of twelve locations and three widths.

The figures themselves can be in one of three standard forms intended for the electronic schematics, normal listings, and compact listings. Otherwise, any old PostScript code can be used for the figures.

Any extra text at the end will get automatically lopped off without any errors getting produced.

The main text makes use of a dozen different fonts. There is that drop cap font, regular sized text in plain, italic and a bold, a slightly reduced text for all caps, again in plain, italic, and a bold, a special drop-centered and bold font for titles, and *Symbol* and *Zapf* for special effects. Additional and seperate font selections are used for your header, footers, and for each of the three illustration styles. The border and the title of each illustration can also be selectively turned off and on.

A unique *keystone* justify is used to "wedge out" the topics in the title block. Similar techniques can be used for wedging left, wedging right, or to flow around any form.

```
My personal horror stories here include your Canadian post office re- fusing to accept my first
class mail, and waiting three hours in a bank for them to decide to use the Canadian exchange
rate for any bank in Nova Scotia. Honest. They couldn't find any country that was named Nova
Scotia, so they had to call up the head office seven times in a row. These epsilon minuses were
about as sharp as five pounds of raw liver.
Finally, they ended up giving me \0334$7.65\0331 for a \0334$24.50\0331 check. Life is too
short for this sort of thing.
Our focus this month is on the electronic lighting controls for rock concerts, discos, theater
lighting, for color organs, and whatever. But first, let's get up to date on . . .
\033c
Library Research
\033d
Several exciting things have been happening at the library lately. First and foremost, lots of
libraries are now putting their card catalogs and public serials lists onto new on-line electronic
\0334BBS\0331 bulletin boards. So, you can now find out what's avail- able without leaving
home.
One example of this would be the \0332Arizona State University\0331 library \0334BBS\0331
reachable at \0334(602) 965-7003\0331.
The second really big news item for all of you hackers is that many libraries are now offering
the great \0332Dialog Information Service\0331. Dialog is a new "supergroup" electronic search
service covering many hundreds of electronic data bases. If the topic that you are reserching is
even remotely popular or scholarly, you will defin- itely find it on Dialog.
While those \0334$2\0331 per minute typical \0332Dialog\0331 charges would seem a tad on
the steep side at first glance, \0334(A)\0331 this is ridiculously and insanely cheaper than
getting the information by any other means; \0334(B)\0331 the searches are far more thorough
and more complete than you could possibly hope to do on your own.\033x

showpage
```

Fig. 5b – Three column gonzo textfile, concluded

Figure five does show you some sample text that illustrates how to use the fancy gonzo template of figure four. While this particular example is from my *Hardware Hacker* reprint series, it is essentially the same (but a tad fancier) than what you are looking at here. A free printed listing of the *Ask The Guru II* is available on phone or letter request.

Figure five has substituted the \033 for any embedded escape commands, just in case your word processor can not handle this detail.

Your print file is around 10K long, and the page makeup time is around thirty seconds using the Apple IIe driving a LaserWriter NTX. This can get reduced to a two second makeup time by doing a pseudocompile that we will get into in a future column. With a compiling pass, the book on demand publication can be done at essentially the "wide open" printing speed of the *LaserWriter*.

I invite you to try printing this any other way by using any other page-making package. If you can even remotely approach the final print speed, flexibility, or quality at 300 DPI, I will give you a free copy of my *Incredible Secret Money Machine* book. Extra points if you are able to do it without using either the *AppleWriter* program or an Apple IIe.

My gonzo justification routines are continually being improved. Recent additions have included an individual character kerning, improved tabbing, and changes that make compiling run time code available.

A free printed listing of the latest gonzo version is now available by writing or calling me per the *need help?* box you'll find at the end of this volume.

The ready-to-run code is also now available for most popular brands of personal computers on my current *Work-in-Progress* disk. We also do custom versions of gonzo that solve specific user problems. Previous examples have included Iowa weather forecasts, exitable and landscapable legal *Diablo* emulators, and for the automatic printing of waranty cards.

Give me a call for lots more info on any and all of this.

Don Lancaster's

ASK THE GURU

April, 1989

Developer resources
The great Pellon ploy
LaserTalk on the Mac
PostScript spirograph
A 57600 baud interface

Jerry Cline and all of the usual suspects over at AZApple are now putting together the third annual *Apple Fiesta*, to get held in the *Safari Resort* in Scottsdale, Arizona on June 2-4. This is a combined Apple II, Macintosh, and Desktop Publishing gathering, and one of the best ones anywhere.

This year's show should be longer and larger than before. Seminars, demos, and hands-on sessions will be given by Guy Kawasaki and Peter Sandys of Apple, Tom Weishaar of *A2 Central*, Roger Wagner of *Roger Wagner Publishing*, Bill Mensch of *Western Design Center* and bunches of similar name-brand people.

I will have a booth there and will be holding several hands-on seminars on PostScript fundamentals and on desktop publishing. Bee will also be giving a seminar or two on book-on-demand publishing and on high tech opportunities for you working artists. Yes, a pre or post-show *tinaja quest* or two could also be arranged.

Plans are also underfoot for a new supergroup regional user group meeting and for several special assembly language conferences. There will also be a *MicroSoft* night, and several pups from *Beagle Brothers* will also be on hand.

The cost of all of the seminars and the exhibits will be around $20 for the whole show. There will be all the usual freebies, prizes, and giveaways as well. Rooms are now available at the special and lower Safari summer rates. Since there is a major car race on the same weekend, early arrangements are a good idea.

The *Sun Remarketing* people have apparently lied to me twice over the phone, causing some serious misinformation to appear in this column. Because of this, they have acheived the coveted *Synergetics* ZZZ- rating, even bottoming the ZZZ rating now uniquely held by *Bell Electronics*.

Sun assured me that they stocked ProDos AppleWriter 2.1 along with all the original manuals. In reality, they have the old and largely useless DOS 3.3 AppleWriter with Xeroxed info sheets instead of real manuals.

Why do I continue to make such a fuss over an older word processor? Because I strongly feel that AppleWriter, with suitable mods, is far and away the finest program available today anywhere for any serious desktop publishing use. On any computer.

An unintentional experiment did tend to dramatically support all my views on this. Seems I started up this beginner's PostScript and desktop publishing course here at EAC. The focus of the course was to have the walk-in students quickly and simply grind out such things as letterheads, business cards, invoices, note paper, advertisements, badges, newsletters, logos, bumperstickers, reports, drawings, resume's, and such. All real world, all here and now. And all done on "bare metal" and host independent word processed PostScript, helped along with a few of my power tools.

The no-nothing and off-the-street beginners were all given a ProDOS AppleWriter 2.1 and a IIe, while the sophisticated power users had their choice of *any* software at all that ran on their choice of Mac, IBM, or even VAX workstations.

The results? All of the no-nothing AppleWriter beginners consistently got faster results and far better results than all the finest Mac, VAX, and 386 power users which our entire Gila Valley had to offer. And they did so having far more fun and with far fewer hassles.

Here is what you need for a decent PostScript word processor: (a) a very powerful and friendly word processor or editor that will work directly with ordinary and unformatted text files; (b) the ability to send and receive all those files as PostScript data from within the same program, all the while receiving the error messages and recording any long code modules sent back from the LaserWriter; (c) the ability to freely and visibly embed font-changing escape commands and other control characters directly

NOTE: Outputs are "unbalanced" at 0 and +3 volts. The max recomended length is five feet. For longer runs and true bipolar "balanced" RS-232, add a single chip and single +5v supply driver such as a Maxim MAX-232.

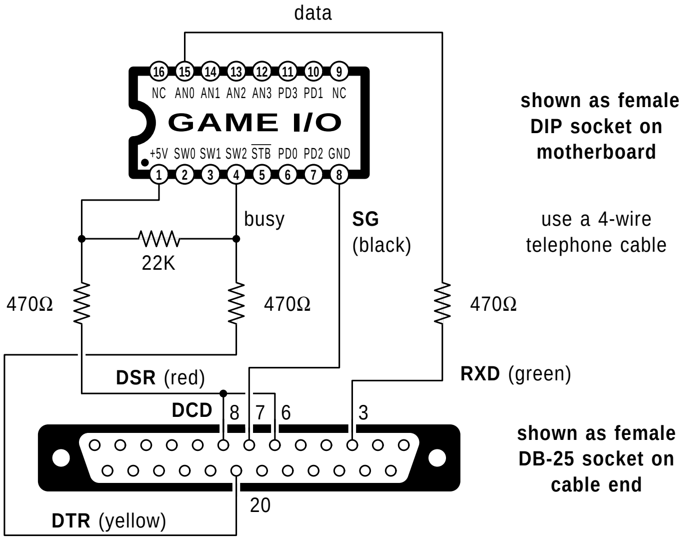
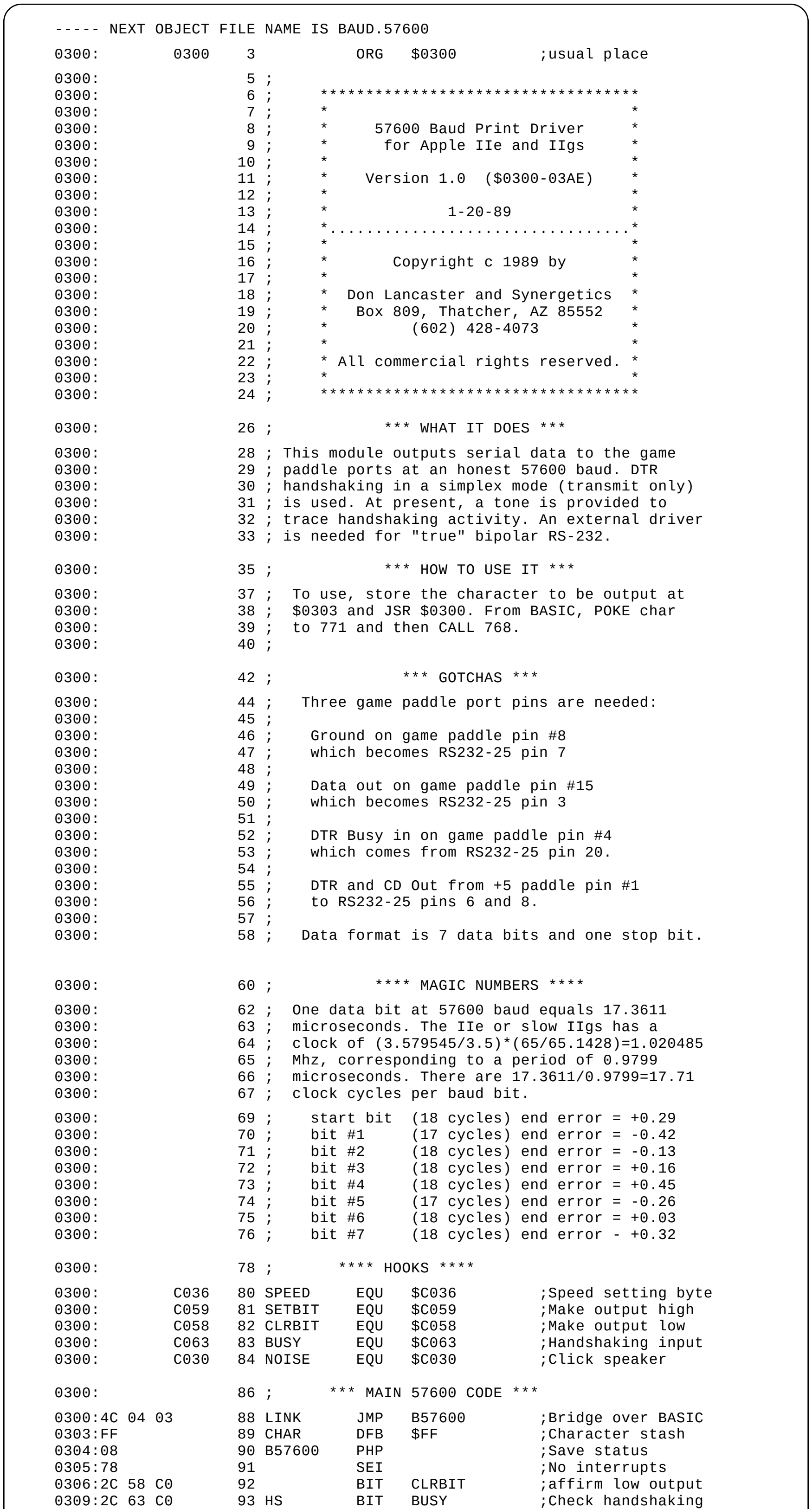

data
16 15 14 13 12 11 10 9
NC AN0 AN1 AN2 AN3 PD3 PD1 NC
GAME I/O
+5V SW0 SW1 SW2 $\overline{\text{STB}}$ PD0 PD2 GND
1 2 3 4 5 6 7 8
shown as female DIP socket on motherboard
busy
SG (black)
22K
use a 4-wire telephone cable
470Ω
470Ω
470Ω
DSR (red)
RXD (green)
DCD
8 7 6
3
shown as female DB-25 socket on cable end
20
DTR (yellow)

Fig. 1 – A game paddle to RS-232 serial adaptor cable.



```
----- NEXT OBJECT FILE NAME IS BAUD.57600
0300:        0300    3          ORG   $0300         ;usual place

0300:                5 ;
0300:                6 ;     **********************************
0300:                7 ;     *                                *
0300:                8 ;     *     57600 Baud Print Driver     *
0300:                9 ;     *      for Apple IIe and IIgs     *
0300:               10 ;     *                                *
0300:               11 ;     *    Version 1.0  ($0300-03AE)    *
0300:               12 ;     *                                *
0300:               13 ;     *             1-20-89             *
0300:               14 ;     *................................*
0300:               15 ;     *                                *
0300:               16 ;     *       Copyright c 1989 by       *
0300:               17 ;     *                                *
0300:               18 ;     *  Don Lancaster and Synergetics  *
0300:               19 ;     *   Box 809, Thatcher, AZ 85552   *
0300:               20 ;     *         (602) 428-4073          *
0300:               21 ;     *                                *
0300:               22 ;     * All commercial rights reserved. *
0300:               23 ;     *                                *
0300:               24 ;     **********************************

0300:               26 ;            *** WHAT IT DOES ***

0300:               28 ; This module outputs serial data to the game
0300:               29 ; paddle ports at an honest 57600 baud. DTR
0300:               30 ; handshaking in a simplex mode (transmit only)
0300:               31 ; is used. At present, a tone is provided to
0300:               32 ; trace handshaking activity. An external driver
0300:               33 ; is needed for "true" bipolar RS-232.

0300:               35 ;            *** HOW TO USE IT ***

0300:               37 ;  To use, store the character to be output at
0300:               38 ;  $0303 and JSR $0300. From BASIC, POKE char
0300:               39 ;  to 771 and then CALL 768.
0300:               40 ;

0300:               42 ;              *** GOTCHAS ***

0300:               44 ;   Three game paddle port pins are needed:
0300:               45 ;
0300:               46 ;    Ground on game paddle pin #8
0300:               47 ;    which becomes RS232-25 pin 7
0300:               48 ;
0300:               49 ;    Data out on game paddle pin #15
0300:               50 ;    which becomes RS232-25 pin 3
0300:               51 ;
0300:               52 ;    DTR Busy in on game paddle pin #4
0300:               53 ;    which comes from RS232-25 pin 20.
0300:               54 ;
0300:               55 ;    DTR and CD Out from +5 paddle pin #1
0300:               56 ;    to RS232-25 pins 6 and 8.
0300:               57 ;
0300:               58 ;   Data format is 7 data bits and one stop bit.


0300:               60 ;           **** MAGIC NUMBERS ****

0300:               62 ;  One data bit at 57600 baud equals 17.3611
0300:               63 ;  microseconds. The IIe or slow IIgs has a
0300:               64 ;  clock of (3.579545/3.5)*(65/65.1428)=1.020485
0300:               65 ;  Mhz, corresponding to a period of 0.9799
0300:               66 ;  microseconds. There are 17.3611/0.9799=17.71
0300:               67 ;  clock cycles per baud bit.

0300:               69 ;    start bit  (18 cycles) end error = +0.29
0300:               70 ;    bit #1     (17 cycles) end error = -0.42
0300:               71 ;    bit #2     (18 cycles) end error = -0.13
0300:               72 ;    bit #3     (18 cycles) end error = +0.16
0300:               73 ;    bit #4     (18 cycles) end error = +0.45
0300:               74 ;    bit #5     (17 cycles) end error = -0.26
0300:               75 ;    bit #6     (18 cycles) end error = +0.03
0300:               76 ;    bit #7     (18 cycles) end error - +0.32

0300:               78 ;       **** HOOKS ****

0300:        C036   80 SPEED     EQU   $C036         ;Speed setting byte
0300:        C059   81 SETBIT    EQU   $C059         ;Make output high
0300:        C058   82 CLRBIT    EQU   $C058         ;Make output low
0300:        C063   83 BUSY      EQU   $C063         ;Handshaking input
0300:        C030   84 NOISE     EQU   $C030         ;Click speaker

0300:               86 ;      *** MAIN 57600 CODE ***

0300:4C 04 03       88 LINK      JMP   B57600        ;Bridge over BASIC
0303:FF             89 CHAR      DFB   $FF           ;Character stash
0304:08             90 B57600    PHP                 ;Save status
0305:78             91           SEI                 ;No interrupts
0306:2C 58 C0       92           BIT   CLRBIT        ;affirm low output
0309:2C 63 C0       93 HS        BIT   BUSY          ;Check handshaking
```

Fig. 2a – EDASM source code for a 57600 baud serial driver . . .

into a document; (d) a powerful and flexible key glossary system; (e) a supervisory language that gives you auto help screens, accepts multiple user inputs, and automatically writes PostScript code for you, especially font selections with user selectable height and width; (f) the same or a similar supervisor that lets you print any left or right pages in any desired order, and otherwise control the automatic font, dictionary, and prolog downloads and otherwise handle job management; (g) everything open, unlocked, understood, and freely modifiable on any and all levels; (h) forces no unwanted carriage returns, line feeds, or any format commands ever; and finally (i) a total freedom from being forced into unnatural acts with small furry rodents.

I will admit that one of the editor programs over on the VAX did seem to remotely approach AppleWriter. Until, of course, you used WPL for auto-downloads, for anamorphic font selections, and for total print management. Then it was all over but the shouting. No contest.

If you can think of any way at all to meet these needs on a Mac or a 386 that is as fast, as powerful, and as convenient as is ProDos AppleWriter on a IIe, please be certain to let me know. A free *Incredible Secret Money Machine* for your trouble.

Apple Computer is up to all sorts of neat new stuff. First and foremost, they have now at long last come out solidly and unequivocally against any and all forms of copy protection.

In *Apple Direct* for January 1989 they flat out state that not only will they not in any way, shape, or form help out with the development or the debugging of your copy protection scheme, but that they also most definitely don’t want any copy protected program to *ever* be run on *any* Apple machine at *any* time for *any* purpose.

They also have a free new *Just Add Water* book to help you form or improve your own Apple user group. The premier issue of *APDAlog* is also now newly available.

Apple has recently shuffled all of their developer programs. The brand new *Partners* program gives you all the bells and whistles for $750 for the first year. This includes a program and development library, and the ability to buy most products at half off. To be a partner, you must end up actually marketing Apple-compatible products and have a genuine business plan. Existing developers will transfer over when they once again come up for recertification.

Their new *Associates* program is open to anyone but has fewer benefits. First year cost is $500 with the library and $350 without.

Individual software and book bits and pieces are also now available to anyone directly through their *APDA* service. This one is a mere $20 per year and a real bargain.

Finally, there is a brand new *Apple Consultant* program that is open to professionals in this field. Contact them directly for more info.

After several recent "My IIgs blew up!" calls, I thought I'd better repeat this warning once again: Do *NOT*, under any circumstances, attach or remove a disk drive from or to a powered IIgs. This is almost certainly guaranteed to blow up both the drive and the motherboard. Remember also that the IIgs has a half minute rundown time. Leave the power off for at least one minute before attaching or removing a drive.

Note that the printer connections usually can be removed or remade at any time, and you probably can get away with removing or replacing a video or monitor cable under power. But disk drives are a no-no.

This month's horror story involves several epsilon minuses who are now buying up the old LaserWriter boards and destroying them just to get the valuable dynamic RAM chips out of them. This is sort of like all those kids that were stealing accounts receivable disks from businesses and reselling them at a buck apiece. Once again, an old LaserWriter board will drop into an old Laserjet machine, giving you a full PostScript printer for under $1300. See last month's column for details.

Addison-Wesley came out with a new "orange" book called *Real World PostScript*. This one includes font modifications, program design, halftoning, layout tips, color seperations, and several other topics. While it is interesting and useful, it is simply not even remotely in the same league as the red and blue books. It is also a

```
030C:10 FB   0309   94          BPL   HS          ; Hang till handshake free
030E:2C 30 C0       95          BIT   NOISE       ; Whap speaker once
0311:AD 36 C0       96          LDA   SPEED       ; IIgs speed and save
0314:48             97          PHA               ;
0315:29 7F          98          AND   #$7F        ;
0317:8D 36 C0       99          STA   SPEED       ; Force slow speed
031A:AD 03 03      100          LDA   CHAR        ; get character

031D:2C 59 C0      102          BIT   SETBIT      ; generate start bit
0320:48            103          PHA               ; stall for 18 - 11 = 7
cycles
0321:68            104          PLA               ; since 11 are used before

0322:6A            106 AX       ROR   A           ; Get First Data Bit
0323:90 09   032E  107          BCC   AL          ;  and pick high/low
0325:B0 00   0327  108          BCS   AH          ; stall for 3
0327:2C 58 C0      109 AH       BIT   CLRBIT      ; Low if a one
032A:B0 00   032C  110          BCS   AH1         ; stall for 17-14 = 3
032C:B0 08   0336  111 AH1      BCS   BX          ; and exit
032E:EA            112 AL       NOP               ; stall 2
032F:2C 59 C0      113          BIT   SETBIT      ; High if a zero
0332:90 00   0334  114          BCC   AL2         ; stall for 17-14 = 3
0334:90 00   0336  115 AL2      BCC   BX          ; and exit

0336:6A            117 BX       ROR   A           ; Get Second Data Bit
0337:90 07   0340  118          BCC   BL          ;  and pick high/low
0339:B0 00   033B  119          BCS   BH          ;
033B:2C 58 C0      120 BH       BIT   CLRBIT      ; Low if a one
033E:B0 06   0346  121 BH1      BCS   BY          ; and continue
0340:EA            122 BL       NOP               ; stall 2
0341:2C 59 C0      123          BIT   SETBIT      ; High if a zero
0344:90 00   0346  124 BL2      BCC   BY          ;
0346:EA            125 BY       NOP               ; stall for 18-14 = 4
0347:EA            126          NOP               ;

0348:6A            128 CX       ROR   A           ; Get Third Data Bit
0349:90 07   0352  129          BCC   CL          ;  and pick high/low
034B:B0 00   034D  130          BCS   CH          ;
034D:2C 58 C0      131 CH       BIT   CLRBIT      ; Low if a one
0350:B0 06   0358  132 CH1      BCS   CY          ; and continue
0352:EA            133 CL       NOP               ; stall 2
0353:2C 59 C0      134          BIT   SETBIT      ; High if a zero
0356:90 00   0358  135 CL2      BCC   CY          ;
0358:EA            136 CY       NOP               ; stall for 18-14 = 4
0359:EA            137          NOP               ;

035A:6A            139 DX       ROR   A           ; Get Fourth Data Bit
035B:90 07   0364  140          BCC   DL          ;  and pick high/low
035D:B0 00   035F  141          BCS   DH          ;
035F:2C 58 C0      142 DH       BIT   CLRBIT      ; Low if a one
0362:B0 06   036A  143 DH1      BCS   DY          ; and continue
0364:EA            144 DL       NOP               ;
0365:2C 59 C0      145          BIT   SETBIT      ; High if a zero
0368:90 00   036A  146 DL2      BCC   DY          ;
036A:EA            147 DY       NOP               ; stall for 18-14 = 4
036B:EA            148          NOP               ;

036C:6A            150 EX       ROR   A           ; Get Fifth Data Bit
036D:90 09   0378  151          BCC   EL          ;  and pick high/low
036F:B0 00   0371  152          BCS   EH          ;
0371:2C 58 C0      153 EH       BIT   CLRBIT      ; Low if a one
0374:B0 00   0376  154          BCS   EH1         ; stall for 17-14 = 3
0376:B0 08   0380  155 EH1      BCS   FX          ; and exit
0378:EA            156 EL       NOP               ; stall 2
0379:2C 59 C0      157          BIT   SETBIT      ; High if a zero
037C:90 00   037E  158          BCC   EL2         ; stall for 17-14 = 3
037E:90 00   0380  159 EL2      BCC   FX          ; and exit

0380:6A            161 FX       ROR   A           ; Get Sixth Data Bit
0381:90 07   038A  162          BCC   FL          ;  and pick high/low
0383:B0 00   0385  163          BCS   FH          ;
0385:2C 58 C0      164 FH       BIT   CLRBIT      ; Low if a one
0388:B0 06   0390  165 FH1      BCS   FY          ; and continue
038A:EA            166 FL       NOP               ; stall 2
038B:2C 59 C0      167          BIT   SETBIT      ; High if a zero
038E:90 00   0390  168 FL2      BCC   FY          ;
0390:EA            169 FY       NOP               ; stall for 18-14 = 4
0391:EA            170          NOP               ;

0392:6A            172 GX       ROR   A           ; Get Seventh Data Bit
0393:90 07   039C  173          BCC   GL          ;  and pick high/low
0395:B0 00   0397  174          BCS   GH          ;
0397:2C 58 C0      175 GH       BIT   CLRBIT      ; Low if a one
039A:B0 06   03A2  176 GH1      BCS   GY          ; and continue
039C:EA            177 GL       NOP               ;
039D:2C 59 C0      178          BIT   SETBIT      ; High if a zero
03A0:90 00   03A2  179 GL2      BCC   GY          ;
03A2:48            180 GY       PHA               ; stall for 18-7 = 11
03A3:68            181          PLA               ;
03A4:EA            182          NOP               ;
03A5:EA            183          NOP               ;
03A6:2C 58 C0      184          BIT   CLRBIT      ; Begin stop bits
03A9:68            185          PLA               ; get old speed back
03AA:8D 36 C0      186          STA   SPEED       ; and restore
03AD:28            187          PLP               ; get old flags back
03AE:60            188          RTS               ; and exit as subroutine

SUCCESSFUL ASSEMBLY: NO ERRORS
```

Fig. 2b – A 57600 baud serial driver, concluded.

tad disjointed and uneven, due to the multiple authors.

Yes, I do have these in stock if you want one. Speaking of which, I do try to keep Adobe’s red, blue, and green books, and this orange book in stock, along with Apple’s new white *Laser-Writer Reference*. I have also got autographed copies of my own *Ask the Guru*, volumes I and II, my *Hardware Hacker*, volume II, and my *Incredible Secret Money Machine* book available.

As per usual, this is your column and you can get tech help and some off-the-wall networking per the end box. Best calling times are 8-5 week-days, *Mountain Standard Time*.

Also as per usual, we’ve gathered all of those *Names and Numbers* together at the end of this volume.

Let’s speed things up a tad . . .

Show me an honest 57600 Baud Serial Interface.

I have long been disappointed with the dismal IIgs serial interface firmware and its software support. First, the IIgs communicates much slower than the IIe. For instance, a IIe takes only 6 microseconds to find out if a character is ready to be read. That IIgs takes 136 microseconds or so to do the same task in its fast mode.

Second, that IIgs interface is incompatible with pretty near all of the important earlier IIe software, so you have to substitute a Super Serial Card instead if you are to reuse many of your older but gooder routines. And, third, we still are lacking essential print drivers for plain old *Send-PS* and even a properly working *Epson* driver. This is inexcusable.

So, step number one is to flush all of your internal IIgs serial firmware, and substitute a Super Serial Card. This at least will get you up to a compatible 19200 baud, and your de-facto baud rates can end up in the 16000 range, at least when you are using AppleWriter.

But, I wanted and needed much more than this. Just how fast can an unassisted IIe or IIgs output serial data? The suprising answer to this is 14 *Megabaud*. Yeah. Mega, not kilo. You do this by putting your data on the super HIRES screen and using the video connector for the serial output. To do so, though, would be socially irresponsible, because of the extreme chagrin and stress this would cause in the Mac and 386 communities.

Instead, let’s just be content to plod along at a leisurely but honest 57600 baud. That will still end up ridiculously faster than AppleTalk with its heavy software overhead, and much faster than most anything else available on the Mac or a 386 machine. And it opens up some of the really exciting speedup tricks that we will look at next month.

For now, we will use a transmit-only *simplex* mode using hard wire DTR handshaking. You can drop on down to 19200 baud with your super serial card anytime you really need a full two-way comm channel.

1. Get into **BASIC.SYSTEM** and then **CALL -151** to enter the monitor.
2. Enter the following code . . .

```
$0300: 4C 04 03 FF 08 78 2C 58 C0 2C 53 C0 10 FB 2C 30 <cr>
$0310: C0 AD 36 C0 48 29 7F 8D 36 C0 AD 03 03 2C 59 C0 <cr>
$0320: 48 68 6A 90 09 B0 00 2C 58 C0 B0 00 B0 08 EA 2C <cr>
$0330: 59 C0 90 00 90 00 6A 90 07 B0 00 2C 58 C0 B0 06 <cr>

$0340: EA 2C 59 C0 90 00 EA EA 6A 90 07 B0 00 2C 58 C0 <cr>
$0350: B0 06 EA 2C 59 C0 90 00 EA EA 6A 90 07 B0 00 2C <cr>
$0360: 58 C0 B0 06 EA 2C 59 C0 90 00 EA EA 6A 90 09 B0 <cr>
$0370: 00 2C 58 C0 B0 00 B0 08 EA 2C 59 C0 90 00 90 00 <cr>

$0380: 6A 90 07 B0 00 2C 58 C0 B0 06 EA 2C 59 C0 90 00 <cr>
$0390: EA EA 6A 90 07 B0 00 2C 58 C0 B0 06 EA 2C 59 C0 <cr>
$03A0: 90 00 48 68 EA EA 2C 58 C0 68 8D 36 C0 28 60 11 <cr>
```

3. Do a **300LLLL** and compare the listing against figure two.
4. Do a **BSAVE BAUD.57600, A$0300, L$AF** to save the code to disk.
5. To install at run time, do a **BLOAD BAUD.57600**.

Fig. 3 – Hand loading your 57600 baud object code driver.

The process works just fine on a II+, IIe, or IIgs. You use the game paddle output as a RS-232 or RS-423 driver. One possible cable schematic appears in figure one. The maximum recommended cable length is five feet, unless you add suitable drivers. Plans are underfoot to offer this cable or cable kits through the folks at *Redmound Cable*. Give them a call for more details.

Source code for a suitable software driver appears in figure two. Annunciator AN0 is used for data out, since reset of a IIe or IIgs drives AN0 and AN1 low and AN2 and AN3 high. The switch input at AN3 is used to avoid any conflicts with the open-apple and closed-apple keys.

You also do have the option of hearing the data as it is output. This can be handy for debug and speed testing. Details on this are different between the IIe and IIgs. In general, *omitting* the speaker whapping will turn the sound *on* on a IIe and *off* on a IIgs. And *including* that speaker whapping will turn the sound *off* on a IIe and *on* on a IIgs. This has to do with the IIgs speed byte at $C036 unintentionally whapping the speaker on a IIe. Arrghhh.

I chose to use brute force code so that two of the data bits could be a microsecond shorter than the others. This was needed for the most precise possible baud rate timing. The code is fully equalized and will produce negligible jitter. Note that the IIgs is forced into its slow mode during any character output times.

To use your driver, you connect your cable and install the machine language code, usually by BLOADing it. From machine language, you store the character to be output in $0303 and then do a JSR $0300. Your new character will get output so long as the DTR busy line is not low. If the busy line is low, the driver will wait till DTR goes high before continuing. Note that a permanantly low DTR will hang your machine.

From BASIC, you just POKE your character to be sent to 771 and then do a CALL 768. Naturally, there is a horrendous speed mismatch if you pair BASIC with 57600 baud, but this mode is very useful for preliminary testing and debugging.

Figure three does give you the

manual hex dump for your final driver, if you are not now using an assembly system.

The figure one circuit only works with modified RS-232 receivers that can accept 0 and +5 logic states as well as the usual -3 and +3. You can also directly drive RS-423 with a connector change. For a "true" RS-232 interface, you may want to add on a *Maxim* MAX232 driver chip.

We'll see some examples of these alternatives in future issues. We will also see a much fancier and far more powerful internal AppleWriter 57600 baud interface next month.

What is LaserTalk?

I rarely review Mac programs, but for those of you using a Macintosh for desktop publishing, the *LaserTalk* code from *Emerald City Software* is certainly worth a close look.

What this does is set you up in an interactive PostScript environment that lets you work with and debug any portion of any program at any time. A special *preview* mode gives you the equivalent of display PostScript, and returns the actual LaserWriter bitmap image to your screen for later study.

Additional screens are available that let you read dictionaries, view the stack, and snoop around.

Another feature that you might find handy is that entire "red book" command set on-line in full detail. This particular feature requires a Mac with hierarchial directories.

Several of my students and quite a few PostScript developers seem most enthuastic over this program.

But I guess I do have three major problems with this LaserTalk code. First, while that preview bitmap is really returned to your screen, you are prevented from doing anything useful with it. As we saw two months ago, captured bitmaps can give you as much as a 3000:1 speedup in some PostScript routines, especially those involved with perspective, 3-D, or star wars lettering. Thus, the bitmap preview feature does not go nearly far enough, and thus ends up as a serious disservice to potential users.

Second, I do have to ask myself whether I would personally use this program. The answer is "probably not", since it does nothing that I am not already able to conveniently and routinely do using AppleWriter and WPL on my IIe. I always prefer to work with "bare metal" on a machine where everything is open, fully understood, and freely modifiable.

LaserTalk would be just one more locked layer getting between me and what I am trying to accomplish.

Third, and by far the worst, the LaserTalk software licensing agreement is so laughingly and ludicrously absurd to the point where no sane individual would ever purchase this program. The agreement would not fit on the disk envelope, so they had to enlarge to envelope to hold it.

The thinking behind this software licensing agreement obviously came from spending many long hours in the outhouse alone.

Very interestingly, though, there is a hidden two-word subliminal message buried in the software licensing agreement. Most potential users can clearly receive this message from ten feet away in two seconds flat.

What is the Pellon Ploy?

Hmmm. Whaddaya know. If you scrape the bottom of the barrel long enough, you will find a secret hidden compartment chock full of goodies.

As most of you are aware, we've looked at dozens of different "magic" materials and processes that greatly extend the possibilities and opportunities of laser printing.

Among others, we have looked at printing on aluminum, creating giant rubber stamps, doing self-numbering tickets, specialty papers, glow-in-the-dark bumperstickers, the *Kroy Kolor* process, transparent materials, those magic *Coburn* materials, the instant transfer decals, and even replicating three dimensional objects in ultraviolet curable plastics.

Full details appeared in the back issues and in my *Ask the Guru I* and *Ask the Guru II*.

At any rate, *Pellon* is a non-woven fabric you will find at your cloth or notions store. It costs a dollar a yard, and sometimes goes by its generic name of *non-woven interfacing*. You will find several weights available.

Believe it or not, the Pellon can be safely run through your *LaserWriter* or other laser printer. You do have to manually feed it and carefully pre-cut your Pellon to exactly 8-1/2 by 11 before you attempt this. The medium thickness work best. Using an SX engine is recommended over the CX one because of its straight paper path.

While the final toner image is not outstanding, it is certainly clear and quite legible.

Why bother? Well simply because

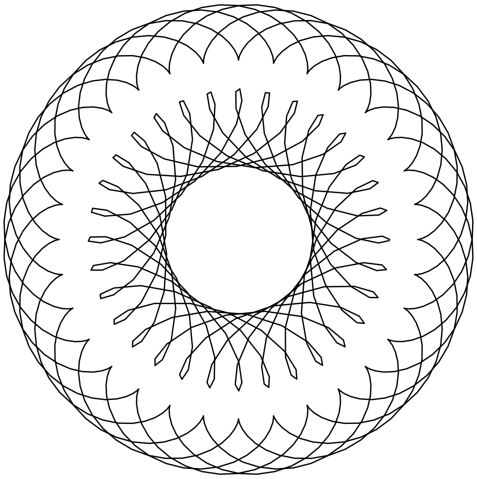


Fig. 4 – The ultimate bagel.

simply because Pellon makes a very good universal pattern and transfer material. It's much more durable than paper. You can also dye it, color it, wrap it in cloth, embroider it, dip it, or paint it.

In a few minutes, my beginning PostScript class came up with over a hundred off-the-wall uses for Pellon transfers. Included were such things as the storyboard objects for child or church show-and-tell sessions, stick-ons, quilt patterns, a giant stencil, enlarged *Zapf* dingbats, display signs, art teaching aides, big pillows with names on them, interior decorating gee gaws, and bunches more.

Ferinstance . . . Say you wanted some big pillows having the words "LEXIE" or "CASSIE" on them. You could start start off with 400 point *Helvetica* bold and print all of the individual letters onto Pellon. Those Pellon letters are then cut out and traced onto a somewhat puffy foam-backed felt material of the type that does not have to be turned under or hemmed. This unusual material, in a contrasting color to the main pillow material, then gets zig-zag sewn (or appliqued) in place.

Obviously, Bee wrote the previous paragraph. Undecipherable technical gibberish. But both Lexie and Cassie really seem to like their pillows.

You might just want to build up some favorite Pellon alphabets and number sets all ready for future use. Thanks to your laser printing, lots of different sizes and styles are easily done. Any way you like.

Use a bigger point size than you think you will need. Remember that points are measured from the *top* of a capital "W" down to the *bottom* of the lower case "g" descender.

Oh yeah. One *extremely* important point: There are two different types of Pellon. There's the plain, or "sew in" type, and the bondable or "heat fusable" type that is intended to be ironed in place. Obviously, you want to use the plain or "sew in" type. The heat fusable Pellon ends up a total disaster if it ever hits the heat fusion rollers on your laser printer.

One more time: Be sure to use the plain Pellon. Do *not* ever use the heat fusable or iron on type.

But why don't you tell me instead? For the first of our two contests this month, just dream up an off-the-wall (or on-the-wall, if you prefer) use for PostScript laser printed Pellon.

```
% Copyright c 1989 by Don Lancaster and Synergetics, Box 809 Thatcher,
% AZ 85552. (602) 428-4073. All commercial rights reserved. Personal use
% is permitted so long as this header remains both present and intact.
% A work in progress disk for Apple, Mac, or IBM is available for $39.50.

% Spirograph routines. Emulates epicycloids and hypoclycoids that have a
% variable length pen arm, similar to "super spirograph" toys.

% To use, translate to desired position. Then define stator teeth /a, rotor
% teeth /b, the number of total revolutions /maxang, resolution in degrees
% /res, and the ratio of the pen arm length to the rotor teeth /penarm.
% Then do a drawspiro.

% Use positive (b) values for "inside" hypocycloids and negative b values
% for "outside" epicycloids. penarm normaly ranges from 0 to 1.

/cposn {/ang exch def a b sub ang cos mul a b sub b div ang mul cos b mul
penarm mul add a b sub ang sin mul a b sub b div ang mul sin b mul penarm
mul sub} def

/drawspiro {newpath a b 1 penarm sub mul sub 0 moveto 0 res maxang 360
mul cvi {cposn lineto currentpoint stroke moveto} for} def

% ///// demo - remove before use. /////

200 300 translate 1 setlinewidth

/a 96 def /b 75 def /maxang 32 def /res 10 def /penarm 0.8 def drawspiro
/a 96 def /b -15 def /maxang 32 def /res 1 def /penarm 1 def drawspiro

showpage
```

Fig. 5 – PostScript code for the ultimate bagel.

There will be the usual *Incredible Secret Money Machine* book prizes for the top twenty or so entries, with an all expense paid (FOB Thatcher, AZ) *tinaja quest* for two awarded to the very best of all.

What is This Month's PostScript Utility?

Well, actually, it is not ready yet. I was hoping to give to you another major and secret blockbuster for this month. Namely some sneaky tricks that will blindingly speed up all your PostScript print times to the point where most jobs will print at the full page-feeding speed of your printer, with, believe it or don't, *zero* make-ready or pre-processing time. Sadly, this one has to wait till next month because one or two final details are not yet fully in place.

Instead, lets back way off and have some quick and easy fun by using a simple new PostScript routine.

Remember those *Super Spirograph* toys? Figure four shows you some of the interesting things you can do with spirograph code as written directly in PostScript, while figure five shows you the rather simple code routines that are involved.

All we really have here is a wheel rotating around a second wheel, with some integer ratio set between those two wheel diameters.

If the outside wheel is smaller in diameter than the inside, you will get curley circles;if the outside wheel is much larger than the inside, you get those cusps.

The ratios of the wheel diameters decide the number of the repeats per revolution of the inside wheel. Make a zillion trips around, and you get a negative image with those chords in them. Unlike real spirograph toys, your pen hole location can easily exceed the diameter of your outside wheel for special effects.

With several simple modifications, you can do spirals or work along an arbitrary path. Thus, this simple PostScript code is a major improvement over the original spirograph toys.

So, for our second big contest this month, just show me some variation on the spirograph theme. Preferably printed onto Pellon.

Don Lancaster's

ASK THE GURU

May, 1989

Secret toner refill tools
PostScript error trapper
A serious AppleTalk bug
Mini-DIN 8 serial cabling
Beginners two-up notepads

Be sure and stop in and say hi at the *Apple Fiesta* regional show that's being presented Friday through Sunday June 2-4 in the *Safari Hotel Convention Center* in Scottsdale, Arizona. This combined Apple, Macintosh, desktop publishing, user supergroup, plus an assembly language programming conference will feature great heaping bunches of name brand Apple people and should have a lot to offer for just about everybody.

As in previous years, I'll be doing some PostScript showing and telling, while Bee will have a seminar or two on book-on-demand publishing and one on high tech opportunities for the working artist.

Give Jerry Cline a call at *AZApple* for more details. Or else just stop in.

For those of you in the midwest, Tom Weishaar of *A2 Central* is also putting together a really heavy-duty developer conference in Kansas City, scheduled for July 21-22.

Apple has now finally reworked and rereleased all of the Apple II series tech notes in ProDOS format. Do see your local user group or BBS system for copies. They're also available on disk from *A2-Central.*

The high points here include the AppleWorks file formats and details on the IIe production changes that did trash out the game paddle connector for some uses. The low points include totally ignoring AppleWriter, especially the IIgs patches.

In retrospect, Apple's unconscionable treatment of far and away their best desktop publishing word processor could only have been caused by Lutus really hacking someone off on his way out the door. That much world class stupidity on Apple's part could only have come about through malice aforethought.

A three disk upgrade of the *IIgs Source Code Sampler* is also now available, and is just about essential to coping with the incredible morass involved in any serious IIgs programming. Contact APDA for details.

The free Apple user group videos are getting much better. See "The Making of Pencilman" in particular.

Apple also has a pair of fat new developer packages out. These now include the *Information Exchange Marketing Guidebook*, along with the new *Information Exhcange Technical Guidebook*. Get in touch with Peg Bailey at Apple for more info.

Lots of brand new LaserWriter font sources are now coming out of the woodwork. Some of these have fully cracked Adobe's encryptation and sizing hints and thus are offering outstanding typographic quality.

Actually, the "secret" process isn't nearly as bulletproof as Adobe thinks it is. Any patient seventh grader can sight read the *eexec* files, by using nothing but Adobe supplied tools. Key portions of the hint machinery are sitting pretty much out in the open in *FlxProc* in *internaldict*. Blue values anyone? We have seen the full details on this in back columns and in *Ask The Guru* volume II.

One source of new fonts is *ATF Kingsley*. One very major advantage of ATF's font machinery is that you can intercept it for non-linear transformations. This means you can now do true perspective and "star wars" lettering as much as 3000 times faster than when using Adobe fonts with their forced pixel line remapping.

Their first offerings are mostly calligraphic, intended for awards and announcements. Call ATF's Limell Schneiker for more details.

What will the next generation of LaserWriter's look like? Well, there already is a NTX-J out which has improved handling of dual characters and is intended for use in Japan. Beyond that, look for a new 400 DPI engine sped up by semi-interpreted PostScript firmware and an AMD 29000 RISC engine that does such things as BitBlts and cubic spline calculations directly in hardware.

Look for a special 400 x 800 mode that dramatically improves photographic halftone quality. Possibly a "B" sized version, good for 11 by 17 and possibly a duplex (double sided) printing option. Possibly better hard

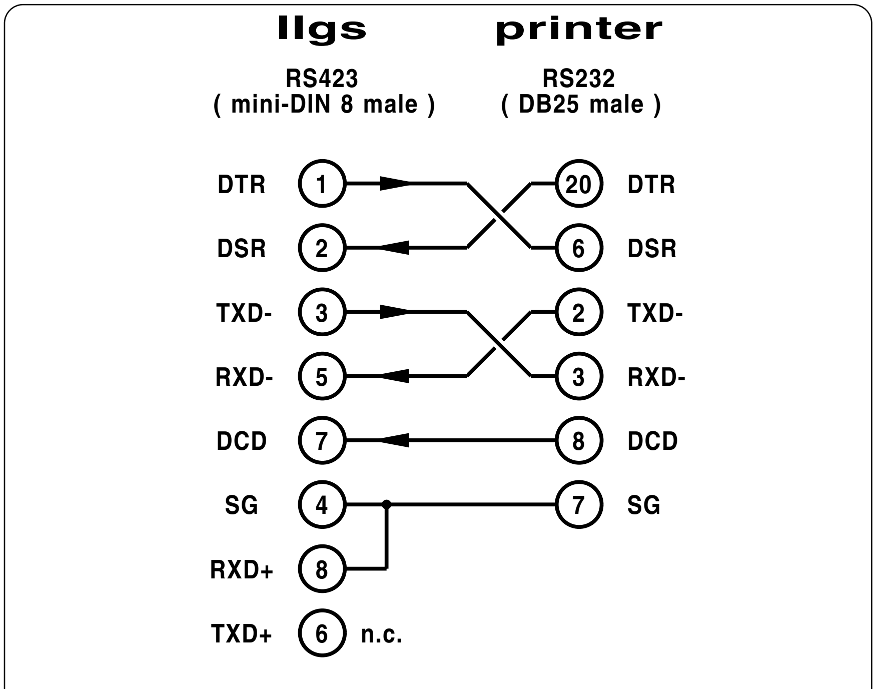


Fig. 1 – A IIgs serial cable for PRINTER use.

hard disk support. Possibly more fonts, perhaps *Garamond* and *Optima* for starters, along with some more *Helvetica* weights.

On the dark side, expect continued outrageously high list prices. Expect the Quickdraw Mafia to continue to try and shove all their non-PostScript non-solutions down your throat. Expect Adobe to continue to refuse to remove the useless and highly demeaning lock on *pathforall*, and still refuse to properly document all of the most useful "hidden" PostScript commands. And expect *Canon* to try draconian anti-refill measures, perhaps by going to a pin-free totally welded package.

For all those of you heavily into electronics, be sure and check my *Hardware Hacker* column over in *Radio-Electronics* this month. It features an easy-to-use full 16-bit digital audio front end for your personal computer or whatever.

Several of my classic hardware books that I do stock autographed copies of here at *Synergetics* include my *CMOS* and *TTL Cookbooks*, my *Micro Cookbooks I* and *II*, and my *Active Filter Cookbook*, and my *Incredible Secret Money Machine*.

As per usual, we also stock *Ask the Guru* I and *II* and *Hardware Hacker II* bound books. Book-on-demand printed with an Apple IIe, of course.

Our biggie this month is a new hard coating process that promises to greatly extend the service life of the *Canon* SX cartridge drums. But, let's first do a summer rerun of "the attack of the killer cables" . . .

How do I Convert from Mini-DIN 8 to DB25?

There are three popular serial communications standards in use today. These are RS232, RS422, and RS423. Each is different in their signal levels and in their choice of connectors.

The older RS232 standard normally uses a DB25 connector and normally has bipolar signal levels of -5 volts for a logic one and +5 volts for the logic zero.

The RS422 standard often may use the DB9 or a Mini-DIN 8 connector. This standard provides for balanced inputs and outputs. As an example, there is a TXD+ output and a TXD- output. These are complementary, so one goes high as the other goes low. There is also a RXD+ and RXD- input pair. Logic is sensed as the differential current *between* these pins.

The RS423 standard is a "single ended" version of RS422 that lets you get by with fewer wires at somewhat lower noise, distance, and speed performance. To convert from RS422 to RS423, you use the TXD+ output and leave the TXD- output unconnected. You then use the RXD+ input and connect the RXD- input to a reference level that is halfway between the one and zero levels.

For instance, if you are running on a single +5 volt supply, the TXD+ and TXD- lines will swing between 0 and +5 volts, so your RXD- input should get connected to a well bypassed +2.5 volt reference. On the other hand, if all your input signals swing from +5 to -5 volts, then you should ground the RXD- input.

It is super important to remember that RXD+ will do absolutely nothing if RXD- is not connected, since the serial receiver works on the current difference present between these two pins. For a logic one, the current must go in the RXD+ line and out the RXD- line. The opposite is true for a logic zero condition.

Most Apple products, including the IIc, IIe, IIgs, the Macs and all of the various LaserWriters have unique serial hardware drivers which can handle any of these three standards. They do this by employing a pair of outputs that swing from +5 to -5 and by using differential input pairs.

Figure one once again shows you how to get from a Mini-DIN 8 to RS-232 and vice versa, by using what Apple calls a *modem* cable, and what the others call a *straight through* cable. On the transmitter end, RXD+ is left unconnected, and RXD- is used as a driver. This handles the negative logic needed by RS232. Over on the receiver end, RXD+ is grounded, while RXD- senses the RS232 levels.

Figure two shows you the "crossed over" connections needed for what the Apple folks call a *printer* cable, and what others call a *null modem* cable. This is by far the more popular of the two connections, as one transmitter will look at the other receiver and vice versa.

On either of these cables, the usual mistakes that everyone will make is forgetting to float the TXD+ output or forgetting to ground that RXD+ input. Once again, it is the differ- ential current between RXD+ and RXD- that gets sensed.

Additional cable variations appear

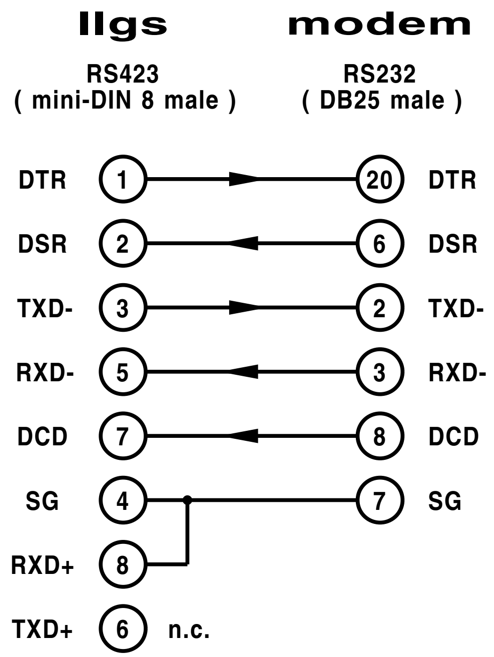


Fig. 2 – A IIgs serial cable for MODEM use.

appear in *Ask the Guru*, volume I. Complete cable assemblies are ready to go at much lower than Apple prices from either *Redmond Cable* or else from the *Microcomputer Cable Company*.

Why Does AppleTalk Badly Foul up the LaserWriter?

Darned if I know. Ask Apple. At any rate, there is an extremely bizarre bug in AppleTalk that can blatently violate PostScript's device independence. And it can drive you up the wall if you do not understand it.

We found it when the Apple, VAX, and the IBM people in my begining PostScript class all were grinding out outstanding typography, while the Mac people were totally trashing all their output. And they all were using absolutely identical work files.

What happens is that the same file sent serially gets treated differently when sent by AppleTalk. That standard and thoroughly documented PostScript serial interface will automatically convert all carriage returns into $10 *newline* characters, while AppleTalk does not.

Which means that custom input scanners will behave differently if the file is sent over AppleTalk rather than over stock serial. Certain older version LaserWriter firmware will also cause the *readline* command to work just fine serially but fail when used over AppleTalk.

The sledgehammer workarounds to this involve never using the *readline* command, or else writing your own input scanner that will automatically convert carriage returns into *newline* linefeeds. This introduces a modest speed penalty into the worst possible place in your code.

The fix to the older copies of my gonzo input scanner is *dup 13 eq {3 sub} if* just before all of the linefeed processing takes place.

Where are all the Secret Toner Reloading Tools?

Since there's two brand new toner cartridge reloading tools available this month, I thought it might be a good time to review all of the secret reloading tools and insider sources.

But first, the new stuff. A great SX cartridge pin puller is now available from *Thompson and Thompson* as their model *AXP43-007.09R Glompen-Stractor*. This one neatly and cleanly pulls the pins with zero damage. The older techniques used here included traverse cutting pliers, screw extractors, woodworker screw starters, or an obscure craftsman tool known as a #8 gimlet.

The really big news, though, is that it looks like SX drum hard recoating is now a reality. In theory, this can greatly extend the SX cartridge life and might eventually reduce or even eliminate the 15:1 per page toner cost penalty of the LaserJet II or either the LaserWriter NT or NTX.

One source of recoating drums and services is Arlin Shepard of *Lazer Products*. Projected recoating costs are in the $8 range. It will be very interesting to see how effective drum recoating eventually becomes.

Let's go back to the old stuff. The best source for detailed maintenence and repair manuals on the CX and SX engines is *Hewlett Packard*. In fact, it is pretty near impossible to intelligently use any Apple LaserWriter without owning the related HP manual that will cover it. The older CX engine (LaserJet, LaserWriter, LaserWriter Plus) manual is #02686-90904, while the newer SX engine (LaserJet II, LaserWriter NT and NTX) manual is part number #33440-90904.

HP has traditionally been a great source for repair and replacement parts for the Apple machines, but lately they have been going to selling the major assemblies only. If the part you want is not individually available from HP, try *Custom Technology* or *Thompson and Thompson*.

As I've mentioned a time or two before, there are two reloading methods, the *punch and go* and the *total teardown*. I overwhelmingly prefer punch and go since it delivers far and

```
/persist true def persist {serverdict begin 0 exitserver} if

/printerror {/$brkpage 64 dict def $brkpage begin /prnt {dup  type/stringtype
ne{=string cvs}if dup length 6 mul /tx exch def/ty 10 def currentpoint/toy exch
def/tox exch def 1 setgray newpath tox toy 2 sub moveto 0 ty rlineto tx 0
rlineto 0 ty neg rlineto closepath fill tox toy moveto 0 setgray show}bind def
/nl{currentpoint exch pop lmargin exch moveto 0 -10 rmoveto}def /=={/cp 0
def typeprint nl}def /typeprint{dup type dup currentdict exch known
{exec}{unknowntype}ifelse}readonly def /lmargin 72 def /rmargin 72 def
/tprint {dup length cp add rmargin gt{nl/cp 0 def}if dup length cp add/cp exch
def prnt}readonly def /cvsprint{=string cvs tprint( )tprint }readonly
def/unknowntype{exch pop cvlit(??)tprint cvsprint}readonly
def/integertype{cvsprint}readonly def/realtype{cvsprint}readonly def
/booleantype{cvsprint}readonly def/operatortype{(//)tprint cvsprint} readonly
def/marktype{pop(-mark- )tprint}readonly def/dicttype{pop (-dictionary-
)tprint}readonly def/nulltype{pop(-null- )tprint}readonly
def/filetype{pop(-filestream- )tprint}readonly def/savetype{pop (-savelevel-
)tprint}readonly def/fonttype{pop(-fontid- )tprint}readonly def/nametype{dup
xcheck not{(/)tprint}if cvsprint}readonly def/stringtype {dup rcheck{(()tprint
tprint())tprint}{pop(-string- )tprint}ifelse }readonly def/arraytype{dup
rcheck{dup xcheck{({)tprint{typeprint}
forall(})tprint}{([)tprint{typeprint}forall(])tprint}ifelse}{pop (-array-
)tprint}ifelse}readonly def/packedarraytype{dup rcheck{dup
xcheck{({)tprint{typeprint}forall(})tprint}{([)tprint{typeprint}
forall(])tprint}ifelse}{pop(-packedarray- )tprint}ifelse}readonly def
/courier/Courier findfont 10 scalefont def/OLDhandleerror errordict
/handleerror get def end errordict /handleerror {systemdict begin $error begin
$brkpage begin newerror{/newerror false store $error /errorname get
(ioerror) ne $error /command get (exec) ne or {vmstatus pop pop 0
ne{grestoreall}if initgraphics courier setfont lmargin 720 moveto (ERROR:
)prnt errorname prnt nl(OFFENDING COMMAND: )prnt/command load prnt
$error/ostack known {nl nl(STACK:)prnt nl nl $error/ostack get aload
length{==}repeat}if systemdict/showpage get exec /newerror true
store/OLDhandleerror load end end end exec}{end end end} ifelse} {end end
end}ifelse} dup 0 systemdict put dup 4 $brkpage put bind readonly put} def
```

Fig. 3 – Adobe's printing and stack dumping error trapper.

delivers far and away the lowest per-page toner costs to the end user. We charge $24 for our local CX and SX reloads. I can still get away with such an outrageously high price since I do live in a rather remote rural area.

At any rate, if you do insist on a total teardown of a CX cartridge, that magic T-10 tamperproof *Torx* bit you'll need is manufactured by *Evco* as their 945B700 set, and can be gotten through *Jensen Tools*.

For punch and go, the best way to produce smooth and truely round reloading and draining holes is to use a #3 *Unibit* from Vice Grip, or one of their imitators. These are once again available from *Jensen Tools* or most any of the large electrical contracting supply houses. One rather good way to replug the holes is with a tapered plastic closure from either *Caplugs* or *Niagara Plastics*

Fusion roller wiper pads should be replaced each time you reload. One source of the custom manufactured peel-and-stick and silicon pressure lubricated Nomex felt strips is *Lazer Products*. These are normally included free with each bottle of their reload toner. A plain old 5/16 inch wood chisel is often the best way to remove the old wiper pad.

Note that washing and reusing wiper pads is a no-no. Their purpose is to deliver a very precisely metered amount of silicon fusion oil. Improperly redone wiper pads can rather dramatically shorten the life of the expensive fuser assemblies.

A good drum lubricant is essential to a proper reload. You can get drum lubricant in bulk from those larger copier repair houses, while small quantities are available as *Pixie Dust*, once again from *Lazer Products*.

Several plastic strips are useful as well. A twelve mil thick piece of the clear butyrate plastic is useful as a feeler gauge for regapping cartridges that have heavy streaking problems. Similar plastic strips can be used for sealing the fresh toner in reloads that have to be shipped somewhere else or stored for long periods of time.

Do let me know if there are any favorite reloading tools of yours that I may have missed.

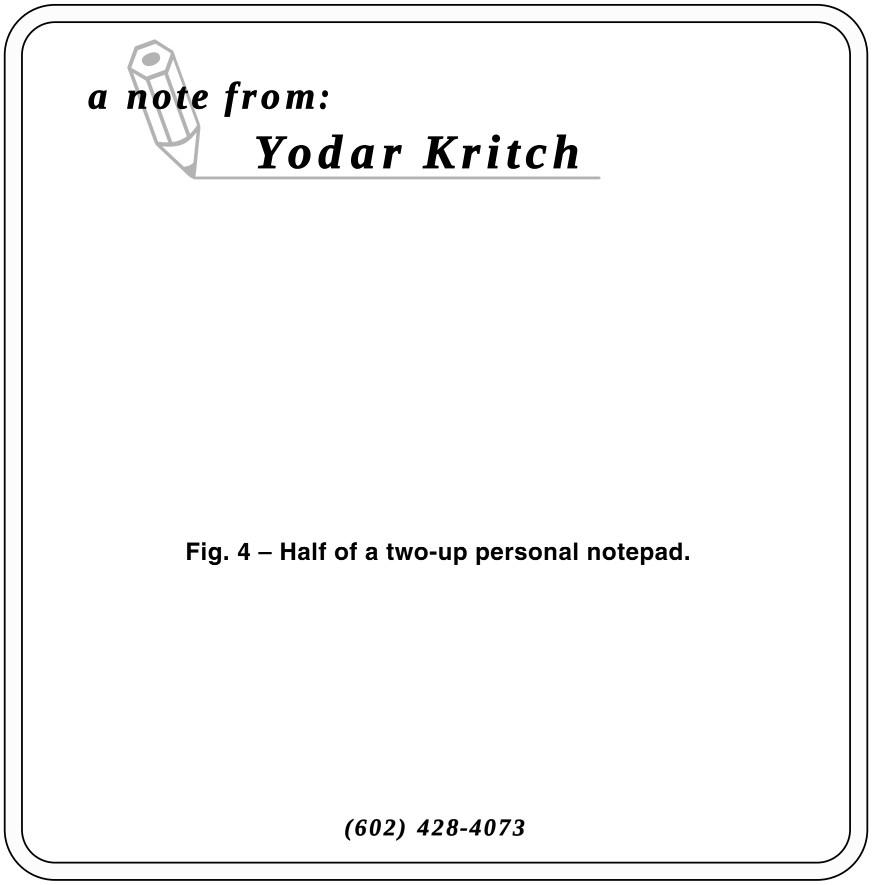


Fig. 4 – Half of a two-up personal notepad.

Tell Me All About Copy Fitting

Both the form and the content of any page layout can be equally important. *Copy fitting* is the arcane art of getting your entire layout to come out even, so it will exactly fit all the space provided.

You are worst off when you are fitting poetry, biblical quotations, or anything else that absolutely has to be word-for-word accurate. You are best off when you're doing your own layout of your own text. In general, though, most average text can be adjusted by adding or removing a few words without hurting the content.

On a five page column layout like this one, at worst you could end up with a ten percent shortfall or excess. And the typical will be more like five percent. And that is only one word in twenty. No big deal.

We have already seen how *post justification* editing can be used on a micro scale to give the most ascetic text lines. Pretty much the same ideas can be used on a macro scale to fit the entire text to the available space.

Text can obviously be made more or less wordy to fit. More often than not, there will be some rather minor or even redundant stuff that can be dropped while actually sharpening the on-page message. Or there will be some important point that might be further amplified or detailed.

So, the very first step is usually to stretch or shrink the text as best you can, trying to sharpen your message while not hurting the content, style, or, above all, the author's intent.

Beyond increasing or decreasing the number of words and all their lengths, PostScript does make it very easy to change the total number of lines per column or lines per page. So one simple ploy is to simply stretch or squash the ledding between lines. A one point difference either way can give you as much as 200 points of adjustment on a three column layout.

A little bit of this will usually be unnoticable. But it is far and away the best to keep the stretch or squash the same for columns on any given page or across any two page spread. And you always want to return to whatever "normal" is just as quickly as you possibly can.

The size and the amount of white space in all your figures can give you the greatest control. Text in figures can be made larger or smaller as needed. You might also stretch or crowd the borders a tad.

Another obvious ploy is to stretch or squash any title or subhead white space. I do this in my *LaserWriter Corner* column which absolutely has to be one *Computer Shopper* page high. The bullets between each topic have programmable white space that automatically stretches or squashes for an exactly fit.

Two useful tricks that can eat up lots of white space are the *sidebar* and the *excerpt*. The sidebar is any words-in-a-box trick that covers an "alike but different somehow" topic that amplifies your main theme or gives some tutorial background.

The excerpt is a two-rule box with a key phrase from the main text inside it in big type. Any excerpts are intended to slow down the browsing reader and draw him into stopping and actually reading the entire text.

Many of the popular page making programs make excerpts very easy to do. Sadly, excerpts end up grossly overused. Excerpts are also a copout, since the real artwork that should go there takes more skill to produce.

If all else fails, you can throw in some innane block of text with a "Tell Me All About Copy Fitting" title or something equally stupid.

What is This Month's PostScript Utility?

February sure was a short month. Which means I didn't quite finish the dramatic PostScript speedup stuff I promised you. So, I thought I would ease off and throw in a couple of lighter topics here. We'll get back to the heavy stuff next month.

By the way, some of the goodies currently stuck in the pipe include that dramatic PostScript speedup that completely eliminates page makeup time; improved curve tracing for such things as engineering graphs and charts; auto form-fillers and auto-invoicers that include spreadsheet capabilities; a powerful badge and bumpersticker generator; an exitable and reenterable Diablo letter/legal portrait/landscape emulator; some improved multiple border procs; new compiling routines; automatic file and system translators; PostScript analog emulations of state variable filters; and great heaping bunches of simple beginner stuff.

Anyhoo, figure three is a listing of the stock Adobe error trapper that prints the page up to the point where the error is made and then dumps the stack. This is also widely available on the various bulletin boards and directly from Adobe. Apparently the code is "not quite" public domain, in that Adobe has copyrighted it yet does seek out the widest possible free dissemenation. A free copy has also been added to my *PostScript Work in Progress* disk as a user service.

Just what can a beginner do with PostScript? My desktop publishing course has given me some suprising answers. What the students get is a word processor and a couple of my invisible "magic" files that have been persistently downloaded.

One of these is my gonzo justify. The second is my PostScript utilities. And the final one is a nuisance dictionary inside the utilities that adds all sorts of easy-to-use commands.

You can get a free printed listing of these when you phone or write. The listings also have appeared in the back issues of *Computer Shopper* and are all available ready to run on my *PostScript Work in Progress* disk.

Figure four shows you a two-up notepad that you might easily put together. This demonstrates how trivially easy "bare metal" PostScript is. The original is in black and a superb high quality gray.

My step-and-repeat utilities can automate the two-up layout process. After printing several dozen pages, you clamp them and run two coats of padding compound along the top edge. A final shearing, and you have a pair of your own high quality and easily customized note pads.

Figure five shows you the trivially simple code involved. Note that my GONZO DL.6, and PS.UTIL.6 must be persistently downloaded before this code can be run.

Several custom mods that all the desktop students did make last night included breaking the pencil line for names with descending characters; substituting hearts, airplanes, and T-squares for the pencil; creating obscene variations; alternate printing using all of the available astrobright colors; and putting a giant light gray Zapf Calligraphic "R" smack in the middle of the writing area.

Please send me your best PostScript variations on a notepad theme. There will be all the usual *Incredible Secret Money Machine* prizes and all-expense-paid (FOB Thatcher, AZ) *tinaja quests* for your best entries.

```
% Copyright c 1989 by Don Lancaster and Synergetics, Box 809 Thatcher,
% AZ 85552. (602) 428-4073. All commercial rights reserved. Personal use
% is permitted so long as this header remains both present and intact. A
% work in progress disk for Apple, Mac, or IBM is available for $39.50.

% Requires persistent download of GONZO.DL.6 and PS.UTIL.6 or later.
% Free printed copies on request.  Note: \033 = embedded escape.

gonzo begin ps.util.1 begin nuisance begin printerror stepnrptparams begin

/twonote [1 2 1 395 0 20 20 true 10 false false] def end

/repeatproc {0 0 10 setgrid % 35 57 showgrid
0 35.5 57 57 1.75 0.8 quickboxpath gsave stroke grestore gsave bestgray
lightgray 0.65 setlinewidth stroke grestore newpath /font1
{/NewCenturySchlbk-BoldItalic findfont [2.0 0 0 2.0 0 0] makefont setfont} def
/font2 {/NewCenturySchlbk-BoldItalic findfont [1.6 0 0 1.6 0 0] makefont
setfont} def /font3 {/NewCenturySchlbk-BoldItalic findfont [1.0 0 0 1.0 0 0]
makefont setfont} def /font4 {/ZapfDingbats findfont [7 0 0 5 0 0] makefont
setfont} def gsave  bestgray 0.8 setgray 1 setlinecap 7.6 50.1 mt line2 17 3.8
55.4 translate -70 rotate 0 0 (\0334/) cl grestore 3 53 (\0332a note from:) cl
10 50.5 (\0331Yodar Kritch) cl 17.7 1.5 (\0333(602) 428-4073) cc} def

(twonote) stepandrepeat
```

Fig. 5 – PostScript code for the two-up notepad.

Don Lancaster's

ASK THE GURU

June, 1989

A low cost jogger
Die cut laser forms
Ultra-fast PostScript
Several GS/OS bugs
Apple-authored books

Apple has corrected many of the really bad bugs in their *ImageWriter LQ* printer. Especially those involving the poor printing at the top and the bottom of the page and some of the severe noise problems. They are even offering a free exchange for any of you that previously bought the older model. This is as close to a total product recall as Apple has ever come.

Word has it that the long delayed IIgs upgrade is due "real soon now". More on this when and where I can tell you all about it.

A+ and *inCider* magazines seem to have merged, with continuity ending up on the *inCider* side. What this means is that the third best Apple magazine and the fourth best Apple magazine, through synergy and their combined resources, should shortly be able to become the seventh best Apple magazine.

My very favorite Apple magazine, of course, is Tom Weishaar's great *A2 Central*, which recently changed its name from *Open-Apple*. Number two remains *CALL A.P.P.L.E.* Even though he ceased publication, back issues and reprints from Bob Sander-Cedarlof's *Apple Assembly Lines* still remain available and contain lots of goodies not available elsewhere.

Two smaller newsletters that are now coming on rather strong are Ross Lambert's *Reboot* for beginning Apple users, and his new *Sourceror's Apprentice* for all you users of Roger Wagner's *Merlin* assembler.

By the way, Merlin is now finally available for the IIgs. Roger sent me a copy of this and many of his other fine programs, but I just haven't had a chance to properly review them yet. In general, Roger does do first rate work, although he and I differ very strongly on whether people who use a mouse to process words are or are not a few chips shy of a full board.

Apple IIe Technical Reference I	*(hardbound, 408 pages, $24.95)*
Apple IIc Technical Reference I	*(hardbound, 576 pages, $24.95)*
Apple IIgs Toolbox Reference I	*(hardbound, 776 pages, $28.95)*
Apple IIgs Toolbox Reference II	*(hardbound, 700 pages, $28.95)*
Apple IIgs Hardware Reference I	*(hardbound, 312 pages, $24.95)*
Apple IIgs Firmware Reference I	*(hardbound, 352 pages, $24.95)*
Technical Into to the Apple IIgs	*(paperback, 160 pages, $9.95)*
Inside Macintosh Volume I	*(paperback, 550 pages, $24.95)*
Inside Macintosh Volume II	*(paperback, 428 pages, $24.95)*
Inside Macintosh Volume III	*(paperback, 280 pages, $24.95)*
Inside Macintosh Volume IV	*(paperback, 326 pages, $24.95)*
Inside Macintosh Volume V	*(paperback, 640 pages, $26.95)*
Inside Macintosh Volumes I-III	*(hardbound, 1240 pages, $79.95)*
Inside Macintosh X-Reference	*(paperback, 128 pages, $9.95)*
Tech Intro to the Macintosh	*(paperback, 320 pages, $19.95)*
Programmer's Into to Macintosh	*(hardbound, 224 pages, $22.95)*
Designing Drivdrs for Mac II & SE	*(hardbound, 288 pages $24.95)*
Macintosh Hardware Reference	*(hardbound, 380 pages $26.95)*
ProDos 8 Technical Reference I	*(hardbound, 208 pages, $29.95)*
Apple IIgs ProDOS 16 Reference I	*(hardbound, 360 pages, $29.95)*
LaserWriter Reference Manual	*(hardbound, 180 pages, $19.95)*
ImageWriter LQ Reference Manual	*(hardbound, 272 pages, $22.95)*
ImageWriter II Reference Manual	*(hardbound, 232 pages, $19.95)*
HyperCard Script Language Guide	*(hardbound, 320 pages, $22.95)*
Human Interface Guidelines	*(paperback, 160 pages, $14.95)*
Apple Numerics Manual	*(hardbound, 320 pages, $29.95)*
BASIC ProDOS Programming	*(hardbound, 296 pages, $29.95)*
Applesoft Tutorial	*(hardbound, 304 pages, $29.95)*
Applesoft Reference Manual	*(hardbound, 368 pages, $22.95)*

Fig. 1 – Some Apple Computer authored books.

At least one of the new SX drum recoating techniques actually does seem to work. A few early reports are showing eight and even nine usable reloads. Contact *Lazer Products* for more details. And competition is at long last driving down the cost of new *Canon* SX and CX cartridges. *Toners Plus* does sell brand new SX cartridges for a mere $79. This is the best price I have seen so far.

I just can not believe that *Hewlett-Packard's* lawyers are letting them get away with all their blatant misstatements of fact in all of their highly misleading "Thou shall not refill" diatribes. Besides being ridiculously cheaper, properly refilled CX and SX toner cartridges can also be considerably blacker, far denser, and much more uniform than the originals.

I have revised all my gonzo justify routines once again. We are now up to GONZO.DL.7 and PS.UTIL.7. The latest features now include improved simple tabbing, individual character kerning, a few bug stomps, and the changes that can give us all of the PostScript compiling tricks we will look at shortly. I've also rewritten the brick wall texture stuff in my perspective utilities to run ridiculously faster than before. Write or call for free printed listings.

Let's see. We gotta cleverly sneak in the usual advertorial here. If you are into any traditional IIc and IIe programming, check into my *Micro Cookbooks*, volumes I and II, my *Enhancing your Apple IIe*, volumes I and II, the *AppleWriter Cookbook*, and my *Apple Assembly Cookbook*. I do have lots of autographed copies on hand here at *Synergetics*. And for PostScript anything, including custom work, just give me a call.

As always, this is your column and

you can get tech help and off-the-wall networking per the end box. Also as usual, all of the names and numbers are gathered together at the end of this volume.

Our really stupendous biggie this month is that I've finally gotten my PostScript speedup stuff up to where I can tell you about it. Later on, I'll show you how your Mac II or even a lowly 386 might now print PostScript almost as fast as you can by using AppleWriter on a IIe.

Lest you scoff at this, please note that I now am doing book-on-demand printing of the typical three column, 6000 character gonzo justified pages including two figures, a header and footer, with an apparently *zero* page makeready time. The successive self-collating pages pour out of my *NTX* printer at the full tilt 8 PPM rate for the entire book.

But first . . .

What Apple Books Are Available?

Figure one can give you a quick rundown of the more popular and easier to find of the Apple Computer authored books. Many are published by *Addison-Wesley*. While I do stock the white LaserWriter Reference here at *Synergetics*, you will find most of the others at most larger bookstores.

For newer and more specialized Apple software and books, check out A.P.D.A for a complete listing.

Two other interesting new Apple publications are their free *Vertical Markets Directory* and their ongoing *Community Affairs News*.

Are There Any More GS/OS Bugs?

Lots of them. In fact, it is getting so bad that I have totally run out of adjectives which might adequately describe this ongoing travesty. So, for our first of the three contests this month, please send me a fresh supply of suitable GS/OS adjectives. Please obey all postal obscenity regulations. There will be all the usual *Incredible Secret Money Machine* books for the top twenty, along with an all expense paid (FOB Thatcher, AZ) *tinaja quest* for two for the very best of all.

We have already seen how GS/OS prevents you from ever resetting into an application and even encourages disk trashing on arbitrary resets.

Another bug involves a misprint in the original GS/OS documentation. It turns out that the page zero memory locations $5A through $5F are very much reserved for internal GS/OS use. These are most definitely *not* available for any printer driver use, despite the early documentation.

The really bad GS/OS bug of the month, though, is that Apple requires you to sign the papers that you are certifiably insane before you do use GS/OS to copy a 5-1/4 inch disk to a 3-1/2 inch one. The GS/OS system, believe it or not, *demands* that you always use an *unlocked* source disk. Otherwise, it will refuse to make the copy and returns an error message back to you.

Outside of the obvious solution of avoiding GS/OS like the plague, the only sane workaround I can think of is to initally copy your 5-1/4 locked original on to a new 5-1/4 unlocked intermediate. Then, use this unlocked intermediate disk to do the actual copy to your 3-1/2 inch disk.

Is Jogging Really An Aerobic Exercise?

Not really, especially if you can avoid it. A *jogger* is a beastie that some epsilon minus of a printing equipment salesperson will charge you $300 to $800 for. Jogging is used to shake pages down so they can go into a binder or can otherwise end up with smooth edges. Properly done, jogging will eliminate most if not all unglued pages, and can even eliminate the need for an edge trim on some jobs. Jogging is just about essential when you are padding your perscription forms, notepads, or a calendar. And machine jogging is far more fun than forever whapping the page edges with a spoon.

Uh, it seems I was in K-Mart the other day, and they are selling *Black and Decker* Model #7448 finishing sanders for $24 or so.

Now, just ask any printing equipment salesperson, and they'll gladly tell you that one way you can build a $24 sander is to take their $600 jogger and remove their paper tray and replace it with a piece of sandpaper.

On the other hand . . .

Any More Desktop Publishing Loose Ends?

Lots of em this month. *Xerox* has a new high end copier called the Model 50 which has an interesting binding system built in. It is apparently based on those old Cheshire heat tape systems, and their supplies and method should be rather easy to adapt to low end binding needs.

The only little problem is that the highly touted Xerox customer sales and service simply does not work. I have sent them over a dozen calls and letters and all I get back are bunches of the cute little cards asking me to please tell them how impressed I was with their previous non-responses.

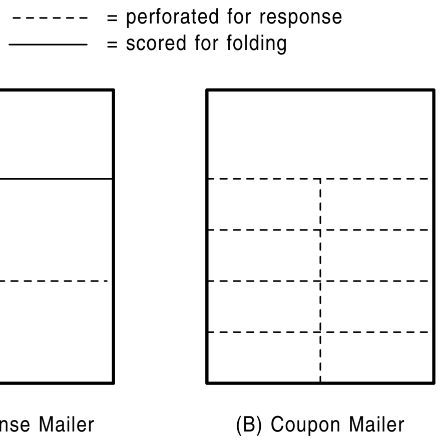


Fig. 2 – Two laser-printable Die-O-Perf products.

As before, the free Kroy Kolor samples are available through Randy Bailey at Kroy, while those low cost substitutes for the pricey Kroy fusion machines are available from Arlin Shepard over at *Lazer Products*.

Kroy also has another unique new product called *TouchDown*. This one starts out with your own laser printed transparency and then creates instant transfers in black, red, blue, or green. These are tougher than the old 3-M INT materials, but not quite as good as die cut vinyl letters. More on this when I get a chance to test it out.

Meanwhile, for any of you that want to get in ahead of the hoarders, their #2100900 starter kit does list for around $79.95.

There is a free catalog cleverly disguised as a shopper's newsletter called *The Printer's Shopper* which lists an incredible variety of printer and printshop accessories, especially stuff of interest to beginning desktop publishers. Their prices are not all that great, but they are definitely better than some.

Die cutting on a laser printer? Why not? We'll have a complete roundup on this interesting topic someday. For here and now, though, check out the *Die-O-Perf* people who sell all sorts of pre-cut blanks that are suitable for laser printing.

Two of my favorites are shown to you in figure two. The unique return postcard mailers are prescored for a triple fold and pre-perforated for a return card. And the coupon mailers (outstanding for any local business promos) do include eight tear-off coupons. The cost is around a nickle each in smaller quantities.

How About That Blinding PostScript Speedup?

As all you regular readers know, I am very big on PostScript book-on-demand printing, where individual book copies are produced on a when and as needed basis. I am currently doing this on three current volumes, including *Ask The Guru I*, *Ask the Guru II* and *Hardware Hacker II*, and do have a dozen more books in the works that will be soon produced this way. Naturally, I'll be happy to send you autographed samples of these at the going rate.

At least for me, the new book-on-demand concept works and works well. Profitably. Here and now.

For self-collating book-on-demand publishing to really be cost effective, you (1) have to reduce your per-page toner costs to well under a quarter of a cent per page, (2) you really should use a duplex printing engine, and (3) must totally eliminate any and all excessive page makeready time.

We've seen in the past how doing your own toner cartridge reloading can get you within shouting distance of (1) and how the latest of PostScript engines now offer (2). The duplex printing can be faked today simply by making a back and a front pass through the machine. So, the what really needs attacked is (3).

The popular pagemaking programs can take an intolerably long three to five minutes to make up a page. And even my gonzo justify routines will take a half minute or so per page. The obvious question is "What tricks can we pull to speed up any later

1. Let the laser printer run continuously. Every time you power down, the font cache and any persistent downloads will get trashed.
2. Use the latest version PostScript engine and firmware. Each newer version has averaged 30 percent faster than the previous one.
3. Use the fastest possible comm channel. The fastest is the SCSI hard disk on the NTX. Second fastest is an honest and unburdened 57600 baud serial channel. Note that Appletalk, with its heavy software overhead, can often end up significantly slower than 9600 baud.
4. Persistently download all common templates, justification routines, drawing utilities, font calls, and whatever. Avoid ever carrying anything inside a file that can be shared with previous or following files.
5. Bind all procedures when and where appropriate. This can result in a free ten to fifteen percent speedup of much of your PostScript code.
6. Use a two-step process where you first get your images printing exactly the way you want them. Then, use the following steps to speed up reruns for later book-on-demand or other repeat printing . . .
7. Carefully measure your communications and execution times to find the bottlenecks. Use the stopwatch features of my nuisance dictionary shown in Ask The Guru II. Note that the balance between execution and comm time is a continuously moving target that changes with your programming style and your choice of baud rate and engine.
8. Eliminate all, or nearly all unneeded comments, spaces, or pretty printing from your runtime files. Use the shortest possible variable names.
9. Try to become just barely baud rate limited. Typically, most of your code will be execution-time limited and should get compiled or else pseudo-compiled for a speedup. Should some code actually baud rate limit, go the other way and let the printer use fancier (and slower) PostScript sequences to shorten your transmitted files.
10. Compile slow portions of your PostScript code by eliminating any calculations. For instance, the positioning and justification of fancy text needs only be done once during a compiling step. Return the compiled results to your host for recording. During runtime, use only fixed constants for positioning and stretching of text.
11. Rearrange your print-time files so that the all of the main text is printed first, followed by all italic text, followed by all bold text, and so on, so that each needed font only gets selected once and only once.
12. Use other compiling tricks when and where appropriate, such as the converting of pixel line remapped images into scanmap files.
13. Avoid compiling anything that makes your runtime textfile so long that the transmission time significantly exceeds the execution time.

Fig. 3 – PostScript runtime speedup secrets.

printing of PostScript jobs?

I have looked into all this rather thoroughly lately and have come up with a few surprising results. The bottom line is that PostScript itself is very much faster than most people suspect. The apparent slowness is caused by poor communications, by poorer programming, and by forcing PostScript to do things over and over again that really only need done once or don't even need done at all.

At present, I can reprint a typical 6000 character, three column gonzo justified text with headers, footers, and two fairly complex figures at the "full tilt" 8 PPM printing time of the LaserWriter NTX. Page after new page after new page. Forever.

It still does require two or three seconds to make up a complete page. But this can be the same two or three seconds of the feeding time available during a normal print cycle. So you can get a virtually zero page make-up time for many typical pages. And those that are not typical can still be printed surprisingly fast.

I've summarized some of the key speedup secrets in figure three. The first five tricks might apply to any PostScript file, at any time, on any machine. The rest of the list applies specifically to pages or files that you know you will want to be reprinting a number of times in the future.

The centermost secret is to get all your communications under control. Most communications on most computers take place far slower than you think they do, owing to the vastly excessive "onion effect" of layer upon layer of controlling software.

The key rule here is to think "two tin cans and a string" for your comm channel. Anything else will slow you down and will severely penalize you when you are doing book on demand publishing. Remember that it all has to get inside the printer before most of it can be used.

If you can afford one, the add-on SCSI hard drive for the NTX should give you the fastest possible comm times. What you do is download all of your files onto the hard disk and then transfer them as needed with a suitable supervisor routine. I have yet to do this, though, and I do suspect things won't end up quite as fast as they really should. More on this in a month or two.

I'm currently using a 57600 baud serial communications channel that goes out the game paddle port of my IIe. Because this uses "bare metal" machine language code, the overhead of getting and sending the characters is essentially zero. Thus, your real baud rate is a true 57600, including the character grabbing times and *all* overhead and formatting.

We saw one 57600 baud driver two months ago, and I'll show you some internal AppleWriter drivers just as soon as I've completely tested them.

What about AppleTalk? The older versions of AppleTalk often end up considerably *slower* than plain old honest 9600 baud when transferring short PostScript files. Especially on the IIgs, but also on those earlier Macintosh machines. Again, this is

```
% Copyright c 1989 by Don Lancaster and Synergetics, Box 809 Thatcher, AZ 85552.
% (602) 428-4073. All commercial rights reserved. Personal use is permitted so long as
% this header remains both present and intact. A work in progress disk for Apple, Mac,
% Atari, or IBM is available for $39.50.

% Intended for use with my GONZO.DL.7. Free printed copies available on request.

% May easily be adapted to any justification program using an unbound awidthshow
% operator and font names of form font0, font1 font2, etc . . .

% Gonzo compile routines  version 7.0  (7 March 89)
% To use, do a persistent download. On file to be compiled, use a

%          startgcompile     - where compiling is to first begin.
%          endgcompile       - if compiling is to be stopped (optional)
%          reportgcompile    - where returned code is to be reported to host for recording

% This program works by opening up a new string dictionary for each font used. The
% awidthshow operator is intercepted and each single font string is then written to the
% appropriate dictionary. When finished, the full dictionaries are returned to the host
% for recording, using a modest compaction that rounds everything to four decimal places
% and eliminates leading or trailing zeros. The 0 yshift and 32 space are suppressed.

% The output is in the form of n1 n2 n3 n4 (text string) z, where n1 is the horizontal
% position in points, n2 is the vertical position in points, n3 is the space stretch, n4
% is the character stretch, (text string) is the message, and z is the decompact proc call.

% The compiled textfile will typically lengthen from 20 to 40 percent, while the run time
% will typically be less than 1/20th that of the uncompiled code.

/startgcompile {/fontsused (0123456789-=) def /dfontname (dfontx) def /glinenumber 1
def /dictsize 500 def /stall {100 {37 sin pop}repeat} def /createfontdicts {fontsused
{dfontname exch 5 exch put dfontname cvn dictsize dict def} forall} def /divertawidthshow
{/awidthshow {dup length 0 gt {dfontname 1 fontn putinterval dfontname cvx exec
glinenumber (        ) cvs cvn mark currentpoint 10 index 8 index 7 index mark exch
{} forall {dup 32 eq {pop}{exit} ifelse} loop ] dup length 0 gt {] put /glinenumber
glinenumber 1 add def}{8 {pop} repeat} ifelse} if systemdict /awidthshow get cvx exec}
def} def createfontdicts divertawidthshow } def

/endgcompile {/awidthshow{ systemdict awidthshow get cvx}def} def

/reportgcompile{/cropzeros {dup length /tslen exch def tslen 1 gt {/tstring exch def tstring
0 get 48 eq {/sst 1 def}{/sst 0 def} ifelse tstring tslen 2 sub get 46 eq tstring tslen 1 sub
get 48 eq and {/tslen tslen 2 sub def} if tstring sst tslen sst sub getinterval} if} def /trunc
{/inum exch def inum dup abs eq {/minus false def}{/minus true def} ifelse /inum inum abs
def inum 0.000001 lt { /inum 0 cvi def}{inum 1000 ge inum 10000 lt and {/inum inum cvi
def}{truncprocs} ifelse} ifelse inum minus {neg} if} def /less1K { /mult 1 def 8 {inum mult
mul 1000 gt {exit}{/mult mult 10 mul def} ifelse} repeat inum mult mul 0.1 add round  mult
div /inum exch def} def /more10K {/mult 1 def 8 {inum mult mul 10000 lt {exit}{/mult mult
0.1 mul def} ifelse} repeat inum mult mul round mult div 0.01 add cvi /inum exch def}
def /truncprocs {inum 1000 lt {less1K}if inum 10000 ge {more10K}if}def /xstrxx ( ) def
/printchar {xstrxx exch 0 exch put xstrxx print flush stall} def /stall {20 {37 sin pop} repeat
} def /stall1 {6 {37 sin pop} repeat} def /dumpstring {(\() print stall flush {dup 92 eq
{(\\13) print pop 52} if dup 40 eq {(\\) print} if dup 41 eq {(\\) print} if dup 127 gt {8 (000)
cvrs (\\) print} {xstrxx exch 0 exch put xstrxx} ifelse print stall1 flush} forall (\)z\r) print
flush stall} def /dumparray {/tarr exch def tarr 0 get trunc 10 string cvs cropzeros print ( )
print flush stall tarr 1 get trunc 10 string cvs cropzeros print ( ) print flush stall tarr
2 get trunc 10 string cvs cropzeros print ( ) print flush stall tarr 3 get trunc 10 string cvs
cropzeros print flush stall tarr 4 get dumpstring stall} def /printfontname {dinuse cvx exec
length 0 gt {(\r) print flush stall stall dinuse 1 5 getinterval print flush stall (\r) print
flush stall} if} def /dinuse (dfontx) def /printgdicts {fontsused {/fxx exch def dinuse 5 fxx
put dinuse cvx exec printfontname length 0 gt {0 1 glinenumber {/dn exch def dinuse cvx
exec dn (    ) cvs cvn known {dinuse cvx exec dn (   )cvs cvn get dumparray} { }ifelse} for}
if} forall } def printgdicts} def
```

Fig. 4 – PostScript gonzo justified text compiler.

because of the onion effect with its "Hi – How are the wife and your kids?" handshaking overhead. Before you blindly use AppleTalk, be sure and compare it against an honest 57600 baud comm direct channel to find out how severe your AppleTalk speed penalty will be.

Regardless of how you communicate, be sure to measure the de-facto "click to clunk" transmission times. You are certain to find bunches of rude surprises along the way.

Once you have gotten the fastest possible transmission times, the next obvious rule is to never send anything over the channel that does not need to be sent. Persistent downloads of such things as justification and drawing routines, templates, and so on, are usually a very good idea. It is usually far better to "fill" your printer with needed routines once in the morning, rather than towing along individual and identical routines with each textfile.

If you definitely know that you will want to reprint your file three or more times in the future, then you can take some additional compiling, calculating and formatting steps that will further reduce all your actual run times. The theory is that a little time spent now can save you bunches in the future.

Before you try this, you should be certain that you do have the *exact* images you really are after. As with any compiler, you must do all of your editing *before* you compile and not after. Get it right first.

In general, you can compile into runtime textfiles by running all your files and then suitably modifying all them with PostScript intercepting and compiling procs, and then returning the modified files back to your host for recording as a new runtime text file. The compiling process can be partially or totally automated. This depends on your programming style and the type of documents you wish to book-on-demand produce.

Several guidelines here. Eliminate all or nearly all of the comments and pretty printing in your run-time files. And let your runtime files work with answers, rather than by doing actual calculations. There is no point whatsoever in making your justification calculations each and every time you

```
% These lines usually go into a persistently downloaded template instead of a header . . .

/font0 {/Times-Bold findfont [54 0 0 54 0 -32] makefont setfont} bind def
/font1 {/Times-Roman findfont [9.75 0 0 9.75 0 0] makefont setfont} bind def
/font2 {/Times-Italic findfont [9.75 0 0 9.75 0 0] makefont setfont} bind def
/font4 {/Times-Roman findfont [9 0 0 9 0 0] makefont setfont} bind def

/z {5 -2 roll moveto 0 32 4 2 roll 0 exch awidthshow} bind def % compile decompacter

%  Here is the actual compiled text. Run it four times back-to-back for a fair test. Your
%  second, third and fourth passes should run at the "wide open" NTX engine speed,
%  provided your computer can communicate as fast an as Apple IIe . . .

font0
60 627.1 0 .2(W)z

font1
116 627.1 1.672 .2(hat is the future of the)z
116 616.6 2.5 .3272(Apple IIgs? Stan Veit)z
116 606.1 .2517 .2(and I have been kicking)z
116 595.6 .924 .2(this around, and we are)z
60 585.1 1.131 .2(getting a few different answers from)z
60 574.6 1.25 .2(several different sources.)z
70 564.1 2.5 .2736(At present, the unit sales of the)z
60 553.6 .4415 .2(IIgs are pretty near the same as those)z
60 543.1 .6003 .2(of the)z
120.6 543.1 .6003 .2(, at four percent of the)z
60 532.6 1.004 .2(total computer market. These figures)z
60 522.1 .858 .2(are from)z
176.9 522.1 .858 .2(, a pretty)z
60 511.6 .7134 .2(much pro-IBM trade journal that is a)z
60 501.1 .01079 .2(great way to keep score of just who is)z
60 490.6 1.25 .2(doing what to whom.)z
70 480.1 2.5 .2424(Apple is sending you conflicting)z
60 469.6 1.073 .2(signals. They have just upgraded the)z
60 459.1 2.5 .2583(IIgs operating system and are now)z
60 448.6 .1191 .2(agressively hiring a big bunch of new)z
60 438.1 2.5 .2932(top-quality Apple II people. Apple)z
60 427.6 1.141 .2(IIc, IIe, and IIgs end user and devel-)z
60 417.1 .6233 .2(oper support is far and away the best)z
60 406.6 1.494 .2(it has ever been. The same goes for)z
60 396.1 2.5 .2938(Apple's own publications and tech)z
60 385.6 1.041 .2(info. Those new "big machine" prod-)z
60 375.1 2.293 .2(ucts from third parties, such as the)z
60 364.6 .9473 .2(improved)z
152.2 364.6 .9473 .2(,)z
71.43 354.1 .7139 .2( and the great new)z
177 354.1 .7139 .2( graphics)z
60 343.6 1.577 .2(environment are coming on-line and)z
60 333.1 1.25 .2(making a big splash.)z
70 322.6 .5474 .2(On the other hand, Apple has been)z
60 312.1 .2283 .2(witholding the IIgs upgrade, and they)z
60 301.6 2.5 .466(purposely crippled that recent IIc)z
60 291.1 .7761 .2(upgrade by keeping AppleTalk and a)z
60 280.6 1.407 .2(real time clock off it. And the intro-)z
60 270.1 0 .2(duction of a Mac)z
151.5 270.1 0 .2( is imminent.)z
70 259.6 1.147 .2(As I see it, there is only one fatal)z
60 249.1 .002357 .2(flaw in the IIgs. And that lies in all of)z
60 238.6 1.136 .2(the monumental costs, the incredible)z
60 228.1 2.102 .2(time delays, rude surprises, and the)z
230 627.1 .3051 .2(inexcusable frustration levels that are)z
230 616.6 2.5 .2693(involved in that)z
326.5 616.6 2.5 .2693( development)z
230 606.1 .6595 .2(environment. If you have to go to all)z
230 595.6 .5284 .2(that innane nonsense, then you might)z
230 585.1 1.197 .2(as well be doing it all on a Mac. At)z
230 574.6 1.141 .2(present, any serious IIgs commercial)z
230 564.1 .13 .2(software development appears to be a)z
230 553.6 1.319 .2(sucker bet because of the sheer frus-)z
230 543.1 1.25 .2(tration problems involving)z
366.2 543.1 1.25 .2(.)z
240 532.6 2.5 .2123(The long term handwriting is on)z
230 522.1 .3816 .2(the wall. Apple now has a secret new)z
230 511.6 1.806 .2(machine in the works known as the)z
299.3 501.1 1.531 .2( that runs both Mac)z
230 490.6 1.25 .2(and IIgs software side by side.)z
```

Fig. 5a – Incredibly fast compiled PostScript text example . . .

time you print. Instead, you do it once during your compiling step.

In a runtime file, you should be able to print anything in any order. So, it makes sense to *only select each font once*. You first print all of your regular text, then all of your italic text, then all the bold text, and so on.

Let us look at an example of compiling some fancy justified text for a dramatic speedup. Figure four shows you that PostScript code I use to compile my gonzo justification routines. You can easily adapt this to any open justification scheme whose *awidthshow* operators can be temporarily unbound.

There are three newer operators here, known as *startgcompile*, *endgcompile*, and *reportgcompile*. Let us look at each one in turn . . .

The *startgcompile* gets used to identify where in your textfile you want to begin compiling. What it does is first create a new dictionary for each font to be used. We do this because we want to sort the all font selections into dictionaries, so that all the bold text can be done at once and so on. Another reason to do this is so that we can report everything at once, rather than having to do so on the fly.

The *awidthshow* operator then gets diverted. Each time the *awidthshow* comes up, we grab the needed parameters of the horiontal position, the vertical position, the char stretch, the space stretch, and the string to be printed and dump them to the correct dictionary for the current font in use. Then the "real" *awidthshow* is used to continue with the original imaging and printing process.

The actual text messages are saved as ASCII character numeric arrays, rather than as strings. This eliminates the tendency of PostScript strings to arbitrarily change as their defined linkings change.

The optional *endgcompile* operator is used if you want to stop compiling at some point in your text. In general, some things should be compiled and some should not. For instance, the present code does not keep track of saves or translates or any wholesale font changes.

When all compiling is completed, and usually at the exit point of your document, you activate *reportgcompile*. This reads the font dictionaries

```
240 480.1 .1388 .2(Actually, if you think about it for a)z

230 469.6 1.573 .2(while, the day of the custom operat-)z

230 459.1 .4004 .2(ing system is nearing an end. Instead,)z

230 448.6 .2462 .2(you just take lots of)z

336.1 448.6 .2462 .2( and a)z

230 438.1 1.472 .2(chip that's running like a bat out of)z

230 427.6 1.407 .2(Cupertino, and then provide suitable)z

230 417.1 1.227 .2(firmware microcode that temporarily)z

230 406.6 1.619 .2(downgrades it into your choice of a)z

253.5 396.1 .7952 .2(, a)z

291 396.1 .7952 .2(, or a)z

340.5 396.1 .7952 .2(. And then)z

230 385.6 .7849 .2(runs anything by anybody. Real time)z

230 375.1 1.25 .2(or even faster.)z

240 364.6 2.5 .2194(Similarly, disk drives are getting)z

230 354.1 1.471 .2(smarter and more flexible. A "multi-)z

230 343.6 1.407 .2(sync" drive that can accept any past)z

230 333.1 .3217 .2(or present media shouldn't be all that)z

230 322.6 2.5 .2611(far away. Interestingly enough, the)z

230 312.1 2.5 .3839(new)z

280.2 312.1 2.5 .3839( operating system does)z

230 301.6 1.244 .2(include a)z

288.5 301.6 1.244 .2(, or)z

250 291.1 .1279 .2( code that, in theory, will let you)z

230 280.6 1.483 .2(read from or to any disk file in any)z

230 270.1 1.25 .2(format, crossing any boundary.)z

240 259.6 2.294 .2(Oh yeah. There is a stupidity on)z

230 249.1 .9998 .2(the new IIgs version)z

362 249.1 .9998 .2( oper-)z

230 238.6 2.153 .2(ating system that is driving all you)z

230 228.1 2.5 .3155(newcomers up the wall. At a first)z

400 627.1 2.5 .2095(glance, it appears as if there is no)z

400 616.6 2.5 .4451(support whatsoever for)z

531.3 616.6 2.5 .4451( inch)z

400 606.1 2.5 .4777(disk drives in this new operating)z

400 595.6 2.5 .3329(system. Actually, the needed)z

400 585.1 2.213 .2(drivers are buried in a subdirectory)z

400 574.6 2.406 .2(named)z

515.3 574.6 2.406 .2(. A copy)z

400 564.1 1.791 .2(of the drivers have to be moved up)z

400 553.6 1.25 .2(into)z

538.5 553.6 1.25 .2(.)z

410 543.1 .3153 .2(Full tech details on)z
```

Fig. 5b - Compiled PostScript text example, continued . . .

in sequential order, and returns only the absolute minimum of information needed to put each message exactly where it belongs on the final page.

Your host must be able to record all this returned information into a suitable textfile. AppleWriter's [Q]-I [esc]-R modem works well here.

Figure five shows you a sample of the actual compiled text returned to the host for recording. The code gets slightly compacted by dropping any trailing spaces, by rounding to four decimal places (which is more than enough for fixed text positioning), by dropping any unneeded leading and trailing zeros and by omitting the always-repeating constants of 32 for the space, the 0 for the y character stretch, and the 0 for the y space stretch. As shown at the beginning of figure five, all of the needed decompacting can be quickly done with a few simple and extremely efficient stack manipulations.

```
400 249.1 2.5 .2889(disk drives, lack of color, and the)z
400 238.6 0 .2(market focus that is far too narrow.)z
410 228.1 1.348 .2(Apple has now released their new)z
536.8 217.6 2.5 .3555(. In-)z
400 207.1 2.338 .2(cluded are complete lists of all the)z
428.3 196.6 2.235 .2( and the)z
505.3 196.6 2.235 .2( commands,)z
400 186.1 1.108 .2(secrets of hard disk access, interface)z
400 175.6 2.073 .2(details, the works. I do have a few)z
400 165.1 .8286 .2(copies in stock here at)z
545.2 165.1 .8286 .2( if)z
400 154.6 1.25 .2(you need one.)z
410 144.1 2.5 .3539(Speaking of which, we do have)z
400 133.6 1.642 .2(autographed copies of volume I and)z
400 123.1 1.79 .2(volume II of)z
516.6 123.1 1.79 .2( in stock)z
400 112.6 1.271 .2(now, as well as volume II of all my)z
475.6 102.1 1.424 .2( stuff. And, if you)z
400 91.6 .4401 .2(need the very latest and the very best)z
400 81.1 .4493 .2(of all my PostScript goodies, do look)z
400 70.6 1.25 .2(into my)z
511.1 70.6 1.25 .2( disks.)z
410 60.1 1.627 .2(As per usual, this is your column)z

font2
99.25 522.1 .858 .2(Computer Reseller)z
102.6 364.6 .9473 .2(AppleWorks)z
158.4 364.6 .9473 .2(PC Transpor-)z
60 354.1 .7139 .2(ter)z
230 501.1 1.531 .2(Brooklyn Bridge)z
307.4 301.6 1.244 .2(File System Trans-)z
230 291.1 .1279 .2(lator)z
464.5 532.6 1.25 .2(APDA)z
464.7 522.1 2.5 .3108(Hewlett-Packard)z
400 217.6 2.5 .3555(LaserWriter Reference Manual)z
400 196.6 2.235 .2(Diablo)z
470.1 196.6 2.235 .2(LaserJet)z
498 165.1 .8286 .2(Synergetics)z
459.2 123.1 1.79 .2(Ask the Guru)z
400 102.1 1.424 .2(Hardware Hacker)z
436.6 70.6 1.25 .2(Work in Progress)z

font4
87.51 543.1 .6003 .2(IBM XT)z
152.7 354.1 .7139 .2(GEOS)z
132.2 270.1 0 .2(K-12)z
305.7 616.6 2.5 .2693(APW)z
345.6 543.1 1.25 .2(APW)z
315 448.6 .2462 .2(RAM)z
364 448.6 .2462 .2(RISC)z
230 396.1 .7952 .2(80386)z
267.5 396.1 .7952 .2(65832)z
317 396.1 .7952 .2(68030)z
252.7 312.1 2.5 .3839(GS/OS)z
272.4 301.6 1.244 .2(FST)z
320 249.1 .9998 .2(4.0)z
335.5 249.1 .9998 .2(GS/OS)z
510.1 616.6 2.5 .4451(5-1/4)z
534.5 595.6 2.5 .3329(5-1/4)z
432 574.6 2.406 .2(/TOOL.DIRECTORY)z
419.9 553.6 1.25 .2(/SYSTEM/SYSTEM.DRIVER/)z
492.4 543.1 .3153 .2(GS/OS)z
539.9 459.1 .7343 .2(H-P)z
471.8 406.6 2.272 .2(III)z
512.6 396.1 2.5 .4509(CD)z
517.8 280.6 1.618 .2(CD-ROM)z

showpage
```

Fig. 5c – Compiled PostScript text example, concluded.

Yes, your compiled text will get longer by twenty to forty percent, compared to the original. But since you gain 20:1 or more in processing time with your compiling step, the final page will print much faster. A lot of textfile overhead also tends to disappear during compiling, partially offsetting the longer results.

As a general rule, you should try and end up just barely baud rate limited. If you are still processing-time limited, then you should try additional compiling tricks. If, on the other hand, your textfiles ever do in fact end up badly baud rate limiting you, then you go back and let your runtime PostScript do more work in exchange for a somewhat shorter file.

Yes, it is a delicate balance. One that depends on you, on your comm speed, the engine, and the version of PostScript you are using. But knowing and keeping that balance can dramatically speed up all your book-on-demand printing jobs.

You might like to use figure five as a timing standard for comparison. To demo this properly, create four repeating copies into a single textfile. Then measure the execution time of the second, third, and fourth pages. Note that each page is being built up from scratch.

If you return to your host before the last page prints, then you will be either PostScript speed limited or are at the "wide open" engine speed. If, instead, you do not return to the host until *after* the last page starts feeding, then you are baud rate limited.

To tell whether you are running at full speed on a NTX, set up a repeat printing of blank pages by using a */#copies 6 def showpage*, open your lower exit tray and make a mark at the previous page edge position at the instant the next page starts its feed.

On the Apple IIe using modified AppleWriter with an internal 57600 baud serial driver, this compiled test file does in fact print at the full speed with zero apparent page makeready. How well does your computer do?