Don Lancaster's

PostScript® Secrets

ISBN 1-882193-45-8

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

In April of 1987, I decided to expand my *Computer Shopper* coverage of the Apple *LaserWriter* line of PostScript-speaking laser printers. What started out as a sidebar to my ongoing *Ask the Guru* column quickly became a new column of its own known as *Don Lancaster's LaserWriter Corner.*

Since editorial space was severely limited, the key operative word here was *intense*. In a cross between "minute manager" and "Just-the-facts-Maam" styles, each month would usually see a few concise use tips ranging the gamut from unique beginner ideas on up to through the utterly gonzo.

There would be a *Material of the Month* that hopefully would lead you into some new and profitable gee-whiz uses for your *LaserWriter*. And a similar *Publication of the Month* which would get you lots of ongoing information. While many of these ended up as free, all of them were thoroughly user-tested.

Finally, each month included a new *Free Font*. I defined a "free font" as any trick you can apply to your already existing fonts to make them do totally new things in uniquely different ways. As you will see, quite a few of these are real mind blowers. All of them are original. Very few have appeared elsewhere.

Despite its name, much of the column's coverage applied to just about any PostScript speaking laser printer and to desktop publishing in general.

I always tried to let the readers in on the real inside stuff – things that *Apple* or *Adobe* or *Canon* or random printing equipment salesmen definitely did *not* want you knowing about.

Secrets such as reducing your toner costs by 15:1 to the 0.3 cents per page range; completely eliminating expensive *AppleTalk* cables; using decent grays instead of the seventeenth most putrid gray that most canned programs force on you; those full color *Kroy* and *Omnicolor* processes; methods to recoat SX cartridge drums; capturing bitmaps back to the host; sight reading *eexec* files (any patient seventh grader can do this); finding useful service manuals and repair parts; the hidden *IWEM* imagewriter emulator; workarounds for the NTX hard disk blowup insidiousities; sources of $1200 LaserWriters; the exact pixel locks for dropout free gray grids; working with random numbers and screens; unique sources for badges, stamps, barcodes, bumperstickers, and padding compound; useful binding systems; and great heaping bunches more.

What you have here are all the original *LaserWriter Corner* columns rewritten, reset, and fully upgraded, in a new layout with a complete index and a new "levelling" of the topics to bring everything up to date.

Incidentally, by the term "*LaserWriter*", I mean a *LaserWriter*, *LaserWriter Plus*, *LaserWriter NT*, or *LaserWriter NTX*. The overpriced, awful print quality, and the totally useless black sheep of the family, that *LaserWriter-SC* is definitely *not* included here, since it by no stretch of the imagination deserves to even get called a *LaserWriter*.

You are holding the sixth in a rapidly expanding series of my book-on-demand self-published books. Each book gets custom and individually printed as it is ordered. Everything you see here was literally beat out on a brick in my own backyard, with some brand new economics that totally revolutionize technical publishing. And each new volume gets a little better than the previous ones.

As always, I do maintain a free PostScript and LaserWriter help line at (602) 428-4073. The best calling times are 8-5 weekdays, *Mountain Standard Time*.

Don Lancaster
Thatcher, Arizona

November, 1989

About the Author

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

Microcomputer pioneer and guru Don Lancaster is the author of 26 books and countless articles. He is considered by some to be the father of the personal computer for his early ground-breaking work with hacker digital electronics and low cost video terminal displays.

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

Other book-on-demand titles in this ongoing series are his *Hardware Hacker*, volume *II*, and the *Ask The Guru,* volumes *I* and *II* reprints.

Don's current software offerings include both his *PostScript Show and Tell*, and his *PostScript Technical Illustrations*, plus numerous companion disks for all his various books. Write or call (602) 428-4073 for a current listing.

Among his popular columns, Don is well known as the *Guru* over in *Computer Shopper*, as the *Hardware Hacker* in *Radio-Electronics* magazine, and as the *Blatant Opportunist* in *Midnight Engineering*.

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

Table of Contents

Don Lancaster's PostScript Secrets

Table of
Contents
1

For more help:
(602) 428-4073

Don Lancaster's LaserWriter Secrets

Table of Contents, continued

For more help:
(602) 428-4073

Table of Contents, continued

For more help:
(602) 428-4073

To cancel your LaserWriter test page on power up, try using . . .

```
serverdict begin 0 exitserver statusdict begin
false setdostartpage end quit  % true restores it
```

But note that this is not nearly as good an idea as it sounds. It is best to let your LaserWriter run continuously, since each reboot trashes the font cache. The test page gives you a quick check on toner and paper quality. On the NTX, a start page is *mandatory* to verify that your hard disk has not blown up again.

Here's a good everyday scroll routine for certificates, etc. . .

```
/bline {gsave  -0.5 0 moveto 0.3 -0.6 0.7 -0.6 1 0 rcurveto stroke
grestore} def

/bloop { gsave  -0.5 0 moveto -1 3 1 3 0 0  rcurveto stroke grestore} def

/bcorner { gsave 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

/drawscrollborder {borderlinewidth borderthickness div 0.37 div
setlinewidth gsave translate borderthickness .37 mul dup scale 2.25
hloops 1 sub 0.0 mul add 3.853 vloops 1 sub 1 mul add translate bcorner
1.3 0 translate hloops 1 sub {bloop bline 1 0 translate } repeat bloop 0.3
0  translate -90 rotate bcorner 1.3 0 translate vloops 1 sub {bloop bline 1
0 translate } repeat bloop 0.3 0 translate -90 rotate bcorner 1.3 0
translate hloops 1 sub {bloop bline 1 0 translate} repeat bloop 0.3 0
translate -90 rotate bcorner 1.3 0 translate vloops 1 sub {bloop bline 1 0
translate } repeat bloop grestore} def

%  /// demo - remove before use. ///

/borderthickness 50 def /borderlinewidth 0.5 def /hloops 34 def /vloops
25 def -90 rotate -792 0 translate % for landscape
37 28 drawscrollborder showpage quit
```

MATERIAL OF THE MONTH –

For real **CERTIFICATE AND BORDER BLANKS**, try *Goes Lithography*. Cost is in the dime range. But be sure to thoroughly vacuum out your LaserWriter after use, since some gold print flecks may remain. From *Goes Lithography* 42 West 62nd Street, Chicago, IL, 60621. (312) 684-6700. Also available retail through the *Paper Plus* chain. Call (213) 436-8291 for nearest location.

PUBLICATION OF THE MONTH –

COMPUTER USAGE FOR SCREEN PRINTING IN EDUCATION is a free *Apple Australia* publication that covers classroom uses of LaserWriters for silk screen printing, printed circuits, on glass etching, fabric printing, business cards, and for color separations. Through *Apple Computer* at 20525 Mariani Avenue, Cupertino, CA 95014 (408) 996-1010.

FREE FONT

Backwards printing can be most handy for store decals, auto stickers, or any other "look through the window" uses. Here are two ways to do it . . .

```
% To completely reverse everything on a portrait page:
   612 0 translate -1 1 scale % use 792 0 for landscape

% To only reverse some lettering:
   /AvantGarde-Demi findfont [-40 0 0 40 0 0] makefont setfont
   300 200 moveto (This is a test) show showpage quit
```

When you are just reversing some lettering, be sure to start at the *right* end with your positioning!

Don Lancaster's PostScript Secrets

1

A fancy scroll border
Certificates & borders
Stopping the test page
LaserWriters in schools
Backwards printing font

For more help:
(602) 428-4073

Don Lancaster's LaserWriter Secrets

Bonus Supplement #1

How to write
and use your own
raw PostScript code

Writing and using "raw" PostScript code...

The key secret to writing and using your own custom PostScript code is to use an ordinary word processor and then set up some two way comm environment between your printer and host. Here's some machine-specific details...

On an **Apple IIe** or the *Apple IIgs*, use *AppleWriter* driving a Super Serial Card. Use your glossary to do a [Q]-I modem receive immediately after sending any file; this gives you a half duplex comm environment. To find your printer status, do a [t]; to reset, *slowly* do a [c] [d]. Initial recommended serial comm parameters are 9600 Baud, 7 data bits, the seldom-used *Space* parity, 1 stop bit, and XON/XOFF handshaking activated. Use of the *AppleWorks* program is not recommended.

On a **Macintosh**, use any old word processor to create your PostScript file. Then save that file as a *standard ASCII textfile* and *not* as a word processor document. If you are using AppleTalk, send and debug your PostScript file using *Send PS* or the *send file* feature of the *Adobe Font Downloader*. If you are using serial comm, use *FreeTerm* or some other suitable two-way communications package.

On an **IBM** or **PC Clone**, you use any old word processor to create your PostScript file. Then save that file in a *standard ASCII textfile* form and *not* as a word processor document. Do *NOT*, under any circumstances use the *copy LPT1* command! Instead, use *Crosstalk* or *ProTerm* to send your file and receive your two-way return error messages. Initial comm values can be 9600 baud, 8 data bits, 1 stop bit, no parity, with the XON/XOFF activated. Use [t] to check printer status, and a *slowly* sent [c] [d] to force a printer reset.

On any **Other Systems**, use a word processor or an editor to create a standard ASCII textfile. Then send that file in a two way comm environment, using [t] to query status and a *slow* [c] [d] to force any printer reset. Start with the IBM comm parameters above. Avoid the use of any packet style network that makes receiving error messages tricky or impossible.

Regardless of your system, it is essential that the LaserWriter is *idle* or "solid green" before attempting to send any file. Later on, you may want to add a printing error trapper to help you with your debugging. These are available in the *Adobe* developer tools disk or in my *PostScript Beginner Stuff* book/disk combo package.

Once your raw PostScript is working to your satisfaction, you can import it into most any applications package by converting it into an *Encapsulated PostScript* or EPS file. Full details on these files appear in the free EPS ap-note from *Adobe Systems* at (415) 961-4400.

For more help:
(602) 428-4073

To automatically collate your *LaserWriter* or *LaserWriter Plus* output, just place your printer on a two-drawer filing cabinet and open the top drawer. The pages will automatically flop over in the right order.

The PostScript command that's used to select landscape printing is . . .

```
-90 rotate -792 0 translate
```

This usually will get placed very near the beginning of your PostScript file, and should *preceed* any other translations, rotations, or scalings.

To lock things to exact one, two, three, or four pixel horizontal boundaries, use one of these commands:

```
currentpoint transform round exch round exch itransform moveto
currentpoint transform 2 div round 2 mul exch round exch itransform moveto
currentpoint transform 3 div round 3 mul exch round exch itransform moveto
currentpoint transform 4 div round 4 mul exch round exch itransform moveto
```

These are handy for fine and dropout-free gray grids or for the elimination of variable widths of thin lines. Vertical pixel locks can also be done. Just eliminate all of the **exch** commands from the above listings.

MATERIAL OF THE MONTH –

The **UNIBIND** system is the best I've found so far for small scale perfect book binding. The covers are clear vinyl with a hot glue strip down them; they are ideal for protecting toner-over-parchment title sheets. Cost is around a dollar per book. The machine is a simple toaster, and is available on free lease if you purchase enough covers. *Unibind*, from *Leonard's Distributors* at 4125 Prospect, Carmichael, CA, 95608. (916) 967-6401.

PUBLICATION OF THE MONTH –

PRINTING IMPRESSIONS is a free middle-of-the-road printer's trade journal with lots of ads for paper, materials, machinery, and services in it. From *Printing Impressions*, 410 North Broad Street, Philadelphia, PA 19108. (215) 238-5300. They also publish many other useful magazines.

FREE FONT

In general, if an *italic* or *oblique* version of a font is available, it should get used. But the *Benguiat* type you see here is not yet available in italic, so you have to fake it. Here is how you can tamper with the third value in the font matrix to get any amount of lean you want . . .

```
/font1 {/Helvetica-Bold findfont [40 0 14 40 0 0] makefont setfont} def
/font2 {/Helvetica-Bold findfont [40 0 10 40 0 0] makefont setfont} def
/font3 {/Helvetica-Bold findfont [40 0 6 40 0 0] makefont setfont} def
/font4 {/Helvetica-Bold findfont [40 0 2 40 0 0] makefont setfont} def

/font5 {/Helvetica-Bold findfont [40 0 -2 40 0 0] makefont setfont} def
/font6 {/Helvetica-Bold findfont [40 0 -6 40 0 0] makefont setfont} def
/font7 {/Helvetica-Bold findfont [40 0 -10 40 0 0] makefont setfont} def
/font8 {/Helvetica-Bold findfont [40 0 -14 40 0 0] makefont setfont} def

0 0 moveto 2 0 font1 (F) ashow 2 0 font2 (R) ashow 2 0 font3 (E) ashow  2 0
font4 (E  ) ashow 2 0 font5 (F) ashow 2 0 font6 (O) ashow 2 0 font7 (N)
ashow  2 0 font8 (T) ashow showpage quit
```

Incidentally, the fake *Benguiat Book Italic* you see here gets done this way . . .

```
/font2 {/Benguiat-Book findfont [9 0 2 9 0 0] makefont setfont} def
```

A much better (although considerably slower) way to handle the longer *star wars* messages appears later in *LaserWriter Secrets #21*.

Don Lancaster's PostScript Secrets

2

Free order reversal
Landscape printing
Printing Impressions
Italic & reverse italic
Horizontal pixel locks
Unibind binding system

For more help:
(520) 428-4073

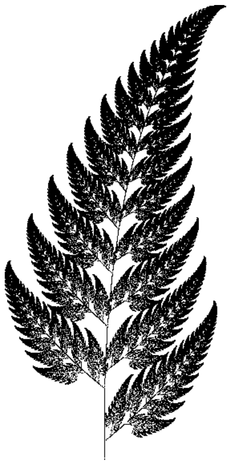
Don Lancaster's PostScript Secrets

Bonus Supplement #2

A PostScript
fractal fern

A PostScript fractal fern...

Here's one of the most stunningly beautiful PostScript images I have ever run across anywhere ever...



```
/problistcreate {mark /counter 0 def probabilities {128 mul round cvi
{transforms counter get} repeat /counter counter 1 add def} forall
counttomark 128 sub neg dup 0 gt { [1 0 0 1 0 0] repeat} {pop} ifelse]
/problist exch def} bind def

/doit {problistcreate 1 1 20 {problist rand -24 bitshift get transform 2 copy
moveto 0.001 10 rlineto} repeat newpath numdots {problist rand -24 bitshift
get transform 2 copy moveto 0.001 0 rlineto stroke} repeat} bind def

% /// demo - remove before use. ///

/numdots 6000 def   % increase for denser image; decrease to print faster

/transforms    [
                  [0 0 0 .16 0 0]
                  [.2 .23 -.26 .22 0 1.6]
                  [-.15 .26 .28 .24 0 .44]
                  [.85 -.04 .04 .85 0 1.6]

               ] def

/probabilities  [ .01 .07 .07 .85 ] def

1 setlinecap
0 setlinewidth
200 300 translate
30 dup scale

doit

showpage quit
```

For a dramatic effect, do this up large with lots of dots. Then Kroy Kolor green on green. Allow three minutes to print an 80,000 dot image on the NTX.

Note that a mere 28 data values determine the *entire* fern image!

Additional details on fractal ferns appear in my *Ask The Guru* reprints, volume II; in *A Better Way to Compress Images* in *Byte*, January 88, pages 215-223; and in Barnsley's *The Science of Fractal Images* from the *Springer-Verlag* press.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

3

Using the font matrix
Metalphoto & *Fotofoil*
Getting repair manuals
Subtle caps & numbers

Instead of the usual method, always use this sequence to select any and all of your Postscript fonts . . .

```
/Helvetica findfont [ 9 0 0 9 0 0 ] makefont setfont
```

This will allow you to squash or stretch the font by setting the character width with the first matrix value and setting the character height with the fourth value.

There's all sorts of italic and rotation tricks you can pull by changing the zeros to something else. For instance, your second zero determines the *climb*, the third the *lean*, the fifth the *horizontal offset*, and the sixth the *vertical offset*.

MATERIALS OF THE MONTH –

METALPHOTO and **FOTOFOIL** are two competing processes that can generate anodized and colored aluminum nameplates, dials, and museum signs directly from your LaserWriter artwork transparencies. Since all the images are actually locked *inside* the metal sheet and are protected by a sapphire overcoating, they are highly vandal resistant. From *Metalphoto*, located at 1835 South Miles Road, Cleveland, OH, 44128 (216) 475-0655 or through *Fotofoil* at 4400 North Temple City Boulevard, El Monte, CA, 91734. (818) 444-4555.

PUBLICATION OF THE MONTH –

The **HEWLETT-PACKARD 2686A SERVICE MANUAL** (#02686-90904, $50) makes an outstanding and easy-to-get *LaserWriter or LaserWriter Plus* repair manual, since nearly all of the CX engine mechanical parts are similar between the two machines. H-P is also a fast and convenient source of replacement parts, albeit an expensive one. And, because of their refusal to provide any decent end user manuals, there is some poetic justice to forcing Apple to pay a $50 cash rebate directly to H-P for each Laserwriter sold. From *HP Manuals*, at 1320 Kifer Road, Sunnyvale, CA, 94086. (800) 227-8164.

The corresponding HP manual for the SX engine based NT and NTX is part number #33440-90904. This one is a "must have".

Sample of NORMAL FONT with internal (123) 456-7890 numbers

Sample of FREE FONT with internal (123) 456-7890 numbers

Yeah, this one is super subtle. But subtle typography tricks become especially important at 300 DPI. Which is one reason that I always will use a four-stage progressive microjustification and hanging punctuation, along with hand-crafted hyphenation, copy fitting, and post-justification editing.

At any rate, any long strings of caps or multiple sequential numerals will look oversize and out of place in any larger text body. The trick is to substitute a second font that is *six to ten percent smaller* for any internal caps or number strings. Here's a simple example . . .

```
/font1 {/Helvetica-Bold findfont [16 0 0 16 0 0] makefont setfont} def
/font2 {/Helvetica-Bold findfont [15 0 0 15 0 0] makefont setfont} def

50 100 moveto font1
(Sample of NORMAL FONT with internal (123) 456-7890 numbers) show

50 50 moveto font1
(Sample of ) show font2 (FREE FONT) show font1 ( with internal ) show font2
((123) 456-7890) show font1 ( numbers) show showpage quit
```

The key test for the correct caps and numbers size to use is "Does it look like it's not there?" This trick becomes super important when narrow columns are fill justified. Note that the intentional use of obviously SMALLER CAPITALS will give you a totally different effect.

For more help:
(602) 428-4073

Don Lancaster's LaserWriter Secrets

Bonus Supplement #3

Using Persistent Downloads

Using Persistent Downloads...

A technique known as *persistent downloading* can make your LaserWriter far more versatile and powerful. With a persistent download you can "permanently" put any favorite fonts, utilities, templates, emulators, dictionaries, or whatever into your printer. These routines will stay in place so long as power is applied. On a later power down, the persistent routines will go away.

For instance, I routinely do a persistent download of my *Guru Powertool Utilities* and my *Gonzo Justify* on first power up in the morning. These will give me my basic working environment for the rest of the day.

To do a persistent download, you just start your PostScript textfile with...

```
serverdict begin 0 exitserver
```

As the file is sent to the printer, your LaserWriter should respond with...

```
%%[ exitserver: permanent state may be changed ]%%
```

More often than not, there will be no page ejected on a persistent download, so an end-of-file, a *quit*, a reset, or a timeout will be needed.

Here's a simple persistent downloading example...

```
serverdict begin 0 exitserver

/blackflash {0 0 moveto 1000 0 rlineto 0 1000 rlineto -1000 0 rlineto
closepath fill showpage } def

quit
```

This is a simple "blackflasher" that will eject a fully black page. Ejecting a fully black page can dramatically improve your camera-ready copy, particularly when fine grays are involved. Just use **blackflash** to activate.

And here are several persistent download WARNINGS...

(1) Persistent downloads use up virtual memory, limiting the size and complexity of your other programs.

(2) Persistent downloads tend to "pile up", so it is a good idea to power down before attempting to download any new ones.

(3) Improperly done persistent downloads can trash the machine so it cannot be used. This is especially true should you change persistent parameters in *serverdict*, most notably the baud rate, other comm parameters, or the password.

On the NTX, a trashed machine can most often be revived by switching to Appletalk and back to 9600 thirty seconds later under power. Baud rate or comm problems can often get corrected by switching to 1200 baud, and then programming the revised comm parameters.

On the LaserWriter and LaserWriter plus, any trashed password requires an extremely complex restoration procedure requiring use of bootleg routines. You can write or call for more info.

The bottom line - **NEVER ALTER YOUR PASSWORD!**

(4) Any persistent routines are best done as closed dictionaries so they do not inadvertently conflict with any other code. Keep your persistent dictionary closed until you need it. Then close it once again after use.

More on persistent downloads appears in the *Apple LaserWriter Reference*, otherwise known as the *White Book*. Call for info.

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

4

New climbing font
Adding a reset knob
Blackflashing secret
Coburn sheet stocks
Graphic Arts Monthly
Save and restore tricks

Your *LaserWriter* drum response is somewhat sensitive to the past few images run over it. The following *black flasher* code can be added to the start of any routine that needs exceptionally good blacks or especially uniform grays . . .

```
newpath 0 0 moveto 1000 0 rlineto 0 1000 rlineto -1000 0 rlineto fill
copypage showpage
```

This does waste toner and paper and, of course, cannot fix any really serious drum or cartridge defects.

Get yourself a 1/4 inch shaft extender from any surplus electronics outfit and add it to the mode switch on your *LaserWriter* or *LaserWriter Plus*. Then add a large knob. Use a two-setscrew shaft extender, or else file a flat on the shaft.

Among its many other uses, this knob can force a reset by switching from 9600 to SPECIAL or vice versa, waiting three seconds, and switching back. But *never* switch to Appletalk during a reset, or you will trash any custom serial settings!

The proper way to do a save/restore is . . .

```
save /snap1 exch def mark
{ all of your wonderful whatever goes here }
cleartomark snap1 restore
```

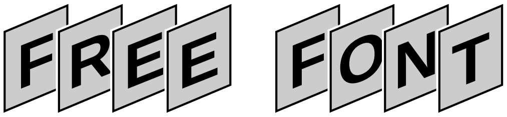
Your internal routine must end any and all of the dictionaries it starts. **clear** and **Initgraphics** are also no-nos, and the code must otherwise behave reasonably. Use snap2, snap3, etc... as needed.

MATERIAL OF THE MONTH –

COBURN offers unique sheet stocks, including prismatics, foils, glow-in-the-darks, pearls, diffraction gratings, and unusual textures. Some are LaserWriter compatible and some are not. From *Coburn*, 1650 Corporate Road, Lakewood, NJ, 08701. (201) 367-5511.

PUBLICATION OF THE MONTH –

GRAPHICS ARTS MONTHLY is yet another free printers trade journal. Its ads are especially heavy on items like menus, invitations, announcements, tags, tickets, thermography, and similar goodies. *Graphics Arts Monthly* is at 249 West 17th Street, New York, NY, 10011. (212) 463-6834.



A climbing font is easily created by suitable tampering with the second value in the font matrix. Here is one possibility . . .

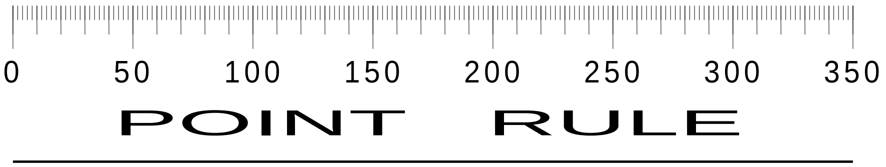
```
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen
/Helvetica-Bold findfont [30 12 0 30 0 0] makefont setfont
/strr (X) def

/climbfont {gsave /msg exch def translate msg {dup 32 ne {strr exch 0 exch
put 0 0 moveto  platewide dup plateclimb mul rlineto 0 platehigh rlineto
platewide neg dup plateclimb mul rlineto closepath gsave 1 setgray
strokewidth 2 add setlinewidth stroke grestore gsave 0.8 setgray fill
grestore strokewidth setlinewidth stroke platewide 2 div strr stringwidth
pop 2 div neg add 10 moveto strr show}{pop} ifelse platespace 0 translate}
forall grestore } def

% /// demo - remove before use ///

/platehigh 38 def /platewide 30 def /platespace 26 def
/plateclimb 0.42 def /strokewidth 1 def

100 200 (FREE FONT) climbfont showpage quit
```

For more help:
(520) 428-4073

Bonus Supplement #4

A PostScript Point Ruler

A PostScript point ruler...



This code generates a PostScript point ruler for you. The ruler is printed diagonally on the sheet so it can be full length and still print on an ordinary letter size page...

```
-30 212 translate -125 rotate

% This routine generates the two unit small ticks...

/xstrt -776 def
/vstrt  200 def
/tick1 -6 def
/pos xstrt def
436 {pos vstrt moveto 0 tick1 rlineto /pos pos 2 add def
0 setlinewidth stroke} repeat

% This routine generates the ten unit medium ticks...

newpath
/pos xstrt def
/tick2 -12 def
88 {pos vstrt moveto 0 tick2 rlineto /pos pos 10 add def 0.20
setlinewidth stroke} repeat

% This routine generates the fifty unit large ticks...

/pos xstrt def
/tick3 -18 def
18 {pos vstrt moveto 0 tick3 rlineto /pos pos 50 add def 0.20
setlinewidth stroke} repeat

% The lettering for all the three digit numbers. Note that there are
% five spaces between each group

newpath
/Helvetica findfont [ 12 0 0 13 0 0 ] makefont setfont /vstrt
vstrt 32 sub def
/xstrt xstrt 12 sub def xstrt vstrt moveto -688 168 moveto
1.66 0 (100     150     200     250     300     350     400     450     500     550
600     650     700     750     800     850) ashow

% The single digit zero...

-734 168 moveto 1.66 0 (50) ashow -780 168 moveto (0) show

% The ruler name...

-465 146 moveto /Helvetica findfont [40 0 0 15 0 0 ] makefont setfont
(POINT   RULE) show

% The bottom line stripe

newpath -776 135 moveto 870 0 rlineto 1 setlinewidth stroke
```

You'll want to cut this out and laminate it in plastic. While useful point rules can be done at 300 DPI, you'll get the best results using a higher resolution printer or phototypesetter.

When combined with an ad message, the point ruler can be a popular and unique sales freebie.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

5

Vertical Fonts
Transfer Magic
Relative layouts
Bakerizing secrets
Translation sequence
APDA Developer stuff

A very sneaky, and zero cost process called *Bakerizing* can make your toner images high gloss, blacker, and far more durable. This is particularly good for business cards and parchment book covers.

Just get a sheet of 1 mil mylar, place it in contact with the original, and heat it with an iron or a *Kroy Kolor* machine. A "completely empty" *Kroy Kolor* sheet will also work and can be reused many times for this calendering secret.

Avoid *ever* using absolute page locations! Instead, use **gsaves** and **grestores** with a translate to "lock" sub units to larger units. This ridiculously simplifies any changes in layout and future reuse.

For instance, if your whatever is inside a box, let **bl** be boxleft and **bb** be the boxbottom. Then do a . . .

```
gsave
bl 10 add bb 50 add translate
{{ your great stuff goes here }}
grestore
```

As you reposition your box or change its size, the contents will automatically track for you. Always work relative, and never absolute.

Watch the sequence in which you **translate**, **rotate** and **scale**, or adjustments and changes will get infuriatingly difficult. For most uses most of the time, you always do your translation first, then a rotation, and finally your scale.

MATERIAL OF THE MONTH –

TRANSFER MAGIC should be available at your cloth or notions store. It instantly converts any toner image into an iron-on applique. You first make a toner on paper copy and stick the *Transfer Magic* to it. Next, you use water to dissolve the paper out from underneath the toner, and then finally iron the applique in place. *Transfer Magic* is supplied through *EZ International* at 130 Grand Street, Carlstadt, NJ, 07072. (201) 935-9005.

PUBLICATION OF THE MONTH –

The **APPLE PROGRAMMER'S AND DEVELOPER'S ASSOCIATION**, or **APDA** for short, is the lowest level of Apple's developer's services. Anyone can join for $24 per year. This gives you a monthly newsletter and a low cost access to all sorts of technical insider publications and software, both *Apple* and third party. One example is their *Asynchronous LaserWriter Driver v4.1* at $14; there are many others. *APDA*, 20525 Mariani Avenue, Cupertino, CA, 95014. (408) 996-1010.

FREE FONT

Here's my approach to a simple vertical font . . .

```
/strr (X) def
/vertfont {gsave /msg exch def translate msg {strr
exch 0 exch put strr stringwidth pop 2 div neg 0
moveto strr show 0 vertspacing neg translate}
forall grestore } def

/// demo - remove before use ///

/NewCenturySchlbk-Bold findfont [ 20 0 0 16 0 0]
makefont setfont /vertspacing 16 def
200 300 (FREE FONT) vertfont showpage quit
```

In general, vertical text looks best inside a border. See *LaserWriter Secrets #26* for border routines that involve the magic of **superinsidestroke**.

FREE FONT

For more help:
(602) 428-4073

Bonus Supplement #5

A Powerful
Step-and-Repeat
Utility

A Step-and-Repeat utility...

Here's a very powerful step-and-repeat that includes sequential numbering, crop or tick marks, portrait -vs- landscape, and lots of other useful features...

```
% Create a dictionary of often-used repeat patterns ...

/stepnrptparams 40 dict def stepnrptparams begin

% The repeat array values, starting from the left are as follows . . .

%  -numpages-     number of pages to be printed
%  -numhoriz-     number of repeats in horizontal direction
%  -numvert-      number of repeats in vertical direction
%  -inchoriz-     horizontal repeat spacing in points
%  -incvert-      vertical repeat spacing in points
%  -horstart-     horizontal starting position in points, lower left
%  -vertstart-    vertical starting position in points, lower left
%  -portrait-     true = portrait;  false = landscape
%  -ticklen-      length of tick in points
%  -ticktrue-     true or false for showing crop marks
%  -seqnumber-    true or false for sequential numbering

/admitonetick [6 5  9 150 60 25 25 true 10 true true] def
/babybumper [1 2 10 270 72 40 30 false 20 true false] def
/badgeaminit [1 2 3 220 220 90 60 false 250 true false] def
/bigbumpstick  [1 1 3 792 205 0 0 true 40 true false] def
/businesscard  [1 3 4 256 143 12 20 true 20 true false] def

/decaapus [1 2 5 306 158 0 0 false 50 true false] def
/hexsplit [1 2 3 306 264 0 0 false 50 true false] def
/lilbumpstick [1 1 5 610 150 0 20 false 60 true false] def
/octopus [1 2 4 306 198 0 0 false 50 true false] def
/quadsplit [1 2 2 396 306 0 0 true 50 true false] def

/readerservice [1 12 25 25 -15 120 450 false 0 false true] def
/seqbuscard [1 3 4 256 143 12  20 true 20 true true] def
/shiplabel [1 1 4 290 180 314 75 false 0 false false] def
/stdplabel [1 1 11 254 74 320 5 false 20 true false] def
/tenlabel [1 2 5 305 144 0 45 false 0 false false] def

/vhsvideospline [1 1 13 424 56 80 35 false 0 false false] def
/3.5disklabel [1 2 3 216 226 100 60 false 20 false false] def
/5.25disklabel [1 1 7 316 110 275 35 false 0 false false] def

end % the repeatparams dictionary

/setrepeatparams {cvn stepnrptparams exch get aload pop /seqnumber exch
def /ticktrue exch def /ticklen exch def /portrait exch def /vertstart exch def
/horstart exch def /incvert exch def /inchoriz exch def /numvert exch def
/numhoriz exch def /numpages exch def portrait {-90 rotate -792 0 translate
} if} def

/onetick { 0 ticklen 2 div rmoveto 0 ticklen neg rlineto ticklen 2 div neg dup
neg rmoveto ticklen 0 rlineto 0 setlinewidth stroke} def

/drawticks {gsave ticktrue {0 0 moveto onetick inchoriz 0 moveto onetick 0
incvert moveto onetick inchoriz incvert moveto onetick} if grestore}def

/stepandrepeat { setrepeatparams numpages {gsave horstart vertstart
translate gsave numhoriz {gsave numvert { drawticks save /rptsave1 exch
def repeatproc  rptsave1 restore seqnumber {/runningnumber
runningnumber 1 add def} if 0 incvert translate } repeat grestore inchoriz 0
translate} repeat grestore showpage grestore} repeat } def

%  //// demo - remove or alter before reuse. ////

/Helvetica-BoldOblique findfont [11 0 0 11 0 0] makefont setfont

/startingnumber 673 def
/runningnumber startingnumber def
/numstring 10 string def

% Your image has to be named /repeatproc and must be well behaved . . .
% Substitute it for this simple demo.

/repeatproc {57 70 moveto  (This is business card #) show
runningnumber numstring cvs show} def

% Your step and repeat format has to be defined in stepnrepeatparams
% Here is how to call it . . .

(seqbuscard) stepandrepeat   quit  % no showpage required (!)
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

6

Arccos and Arcsin
Instant Printer mag
Dictionary overflows
Courier font revisited
Tamperproof *Torx* bits

If you ever get a **dictfull** error message and are unable to change the size of the dictionary that has been overfilled, just do a...

```
600 dict /mynewdict exch def     mynewdict begin
```

Be sure to end this dictionary if you must **restore** or return control to something like a supervisory program.

To pick up Postscript's missing **arcsin** command, try this sequence...

```
/asin {/hypot exch def /xside exch def hypot dup mul xside dup mul sub sqrt
xside exch atan} def
```

It gets used this way...

```
xside hypotenuse asin
```

Similarly, to pick up Postscript's missing **arccos** command, try this sequence...

```
/acos {/hypot exch def /yside exch def hypot dup mul yside dup mul sub
sqrt yside exch atan} def
```

It gets used this way...

```
yside hypotenuse acos
```

Both of these may be further simplified by playing directly with the stack.

MATERIAL OF THE MONTH –

Those hee-hee-haw-haw funny screws on your *LaserWriter* or *LaserWriter Plus* type CX cartridges can easily be removed by using a **#10 TAMPERPROOF TORX BIT**, available as part of the #945B700 set manufactured by *EVCO* at 3451 Lorna Road, Birmingham, AL 35236. (205) 822-5381. *Jensen Tools* at (602) 968-6231 is one stocking source.

PUBLICATION OF THE MONTH –

The **INSTANT & SMALL COMMERCIAL PRINTER** is a free trade journal aimed at low end printshops and walk-in copy houses. It does have lots of useful desktop publishing products and ideas in it. They are heavily into binding systems, tags, tickets, thermography, labels, papers, cutters, drills, etc... Box 368 Northbrook, IL 60062. (312) 564-5940.

FREE FONT

Is their any life after using a fixed pitch Courier? There sure can be, if you ever decide to get sneaky enough about it...

```
/superstroke {save /sssnap exch def /sscmd exch def mark 0 2 sscmd length
2 div cvi 1 sub 2 mul {/aposn exch def gsave sscmd aposn get setlinewidth
sscmd aposn 1 add get setgray stroke grestore} for cleartomark sssnap
restore newpath} def

/novacourier {save /ncsnap exch def /msg exch def translate mark /strx (X)
def 1 setlinecap 1 setlinejoin 0 0 moveto /charnum 0 def msg {strx exch 0
exch put strx false charpath currentpoint strokerule superstroke exch
kernrule charnum get add exch moveto /charnum charnum 1 add def} forall
cleartomark ncsnap restore } def

%  /// demo - remove before use ///

/Courier findfont [80 0 0 80 0 0] makefont setfont

/strokerule [8 0 4 1] def

/kernrule [2 4 3 0 0 0 2 3 0] def % lets you custom kern each character

200 300 (FREE FONT) novacourier  showpage quit
```

The **strokerule** says to stroke 8 points wide black, followed by four points wide white. The numbers in **kernrule** let you trim the individual character spacings. There are 2 points between the F and R, 4 points between R and E, etc.

For more help:
(602) 428-4073

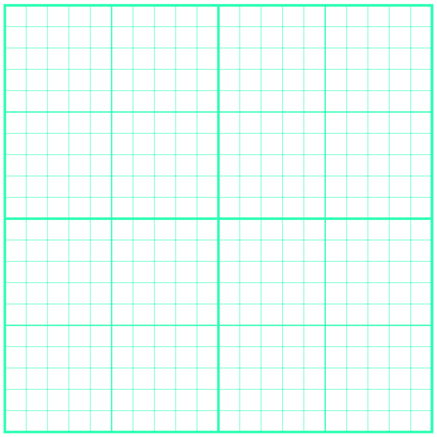
Don Lancaster's LaserWriter Secrets

Bonus Supplement #6

Dropout-Free
Fine Gray Grid
Detailed Example

A dropout-free fine gray grid example . . .

Creating a totally uniform finegray grid that is dropout free is not a trivial task on a 300 DPI *LaserWriter*. Here is some example *rubbergrid* code from my *Don Lancaster's PostScript Beginner Stuff* that shows you how it's done...



```
%  Creates fine gray grids without dropouts or rattiness.
%  The code shown is device specific for 300 dpi printers.

%  To create a grid, use  -hpos- -vpos- -gridsize- setgrid Until restored, all
%  further images will be "locked" to the grid and will expand and contract
%  with it. Note that optimum linewidths and font sizes will usually be much
%  less than 1.0 after locking.

%  To show a grid, use -#hlines- -#vlines- showgrid.

%  The seegrid command displays the grid when true.
%  The fat5 command emphasizes every fifth line when true.
%  the fatter10 command emphasizes every tenth line when true.

/quadpixel {transform 4 div round 4 mul itransform} def

/setgrid {save /rubbersnap exch def quadpixel /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{ 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

%  use examples: -xpos- -ypos- -gridsize- setgrid -#hlines- -#vlines- showgrid
%  {anything you want locked to the grid} clear rubbersnap restore

%  /// demo - remove before use ////

100 200 10 setgrid 20 20 showgrid showpage quit
```

Note that there is precisely one dot at each grid crossing. All linewidths of an intended width are totally uniform. Note also that you are now locked to your grid, with *one* unit matching *one* block on the grid. If you ever want to get off of your rubbergrid, simply do a **clear rubbersnap restore**.

There are several minor gotchas: The grid may not turn out exactly the precise size you want it to. This is the price you do have to pay for uniformity and no dropouts at 300 DPI. The grid size and position may also change slightly if you switch to a higher resolution phototypesetter.

And the grid works best if you do not do any translation or scaling before you call the grid code. If you must translate, be sure to call **quadpixel** to translate only to exact multiples of four pixels.

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

7

Hot stamping foils

Garbage collection

Ask The Guru I & II

Rebuilt LaserWriters

Classic litho chokes

Toner refill supplies

Except for the **restore** command, there is no "garbage collection" as such in PostScript. Which means that every time you re-define something, new virtual memory gets committed, rather than overwriting the previous definition. Two good ways to conserve the available memory include bracketing any sub-job with a **save** and a **restore**, and avoiding ever defining one procedure from within another one that gets called repeatedly. Use your **vmstatus** command.

Here's a few sources of hot stamping foils ...

Bind-it
150 Commerce Drive
Hauppauge, NY 11788
(800) 645-5110

Hoechst Corp
PO Box 1400
Greer, SC 29652
(803) 879-5000

Identicolor
720 White Plains Road
Scarsdale, NY 10583
(914) 472-6640

Lamart
16 Richmond Street
Clifton, NJ 07015
(201) 772-6262

Maple Roll Leaf
2285 Ambassador Dr
Windsor, ONT N9C 3R5
(519) 966-4721

Transfer Print Foils
PO Box 518
E. Brunswick, NJ 08816
(201) 238-1800

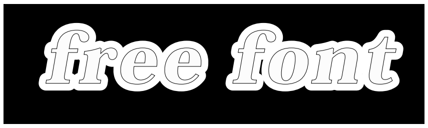
Transfer Print Foils has an outstanding *Foiled Again* newsletter.

MATERIALS OF THE MONTH —

You can get **TONER CARTRIDGE REFILLING SUPPLIES** from either Arlin Shepard at *Lazer Products Inc.*, 12741 East Caley Avenue #130, Englewood, CO 80111, (303) 792-5277; or by way of Don Thompson of *Thompson and Thompson*, 23072 Mullen, El Toro, CA 92630. (714) 855-3838. In addition, Arlin manufactures the fusion wipers and drum lubricant, while Don carries the special teardown tools, low cost rebuilt LaserWriters, and replacement fuser assembly parts.

PUBLICATION OF THE MONTH —

I like to think that my book-on-demand published **ASK THE GURU REPRINTS I & II** have bunches of useful PostScript, LaserWriter, and desktop publishing stuff in them, developed in far more depth than we can do here. From *Synergetics*, Box 809, Thatcher, AZ 85552. (602) 428-4073.



The classic litho chokes and spreads are easily handled by using the Postscript language applied to existing fonts. The secret is the **charpath** operator, which must be called twice to prevent the spread on one letter from trashing the previous one...

```
/Bookman-DemiItalic findfont [50 0 0 50 0 0] makefont setfont
/msg (free font) def /ws (X) def
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen   % a nice gray

/spreadproc { ws false charpath gsave 1 setlinecap 1 setlinejoin 10
setlinewidth 1 setgray stroke grestore currentpoint newpath exch 2 add
exch moveto} def

/letterproc { ws false charpath gsave 0.99 setgray fill grestore gsave 0.25
setlinewidth stroke grestore currentpoint newpath exch 2 add exch moveto}
def

172 272 moveto 0 83 rlineto 292 0 rlineto 0 -83 rlineto closepath fill 200
300 moveto gsave msg {ws exch 0 exch put spreadproc} forall grestore
gsave msg {ws exch 0 exch put letterproc} forall grestore 218 7 rmoveto 0
12 rlineto 5 0 rlineto 0 -12 rlineto closepath 1 setgray fill % a manual patch
showpage quit
```

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #7-A

Carriage return hassles

Solving PostScript carriage return hassles...

With two VERY IMPORTANT exceptions, PostScript permits extra carriage returns or linefeeds just about anywhere. For instance, you can use carriage returns for "pretty printing" to make proc listings more readable. You can add carriage returns to hex ASCII listings to improve formatting. You can add carriage returns to images to make them look better on a host screen.

But ...

You CANNOT put any unwanted or extra carriage returns (INSIDE A POSTSCRIPT STRING), as they will be interpreted and used as REAL carriage returns or linefeeds.

You CANNOT put any unwanted or extra carriage returns INSIDE A BLOCK OF TEXT BEING READ AS A CURRENTFILE. as, once again, they are likely to be intrepreted and used as REAL carriage returns or linefeeds in strings.

Sadly, some editors and many earlier word processors have their heart set on throwing in extra carriage returns, either on an initial load or when formatting what they perceive to be a text line. The usual results will be PostScript listings that get spacier and spacier, or unwanted forced paragraph breaks.

The first and foremost cure is understanding WHEN and HOW your word processor or editor may try to throw in extra carriage returns. Usually there is one or more simple cures.

A VERY IMPORTANT POSTSCRIPT RULE

A carriage return or linefeed that is immediately preceeded by a reverse slash will get ignorted by PostScript.

There are three solutions ...

Best:

(1) DO NOT EVER LET ANY UNWANTED CARRIAGE RETURNS HAPPEN.

Permissible:

(2) If an unwanted carriage return does happen, preceed any and all of them with an immediate reverse slash

(3) Or, throw in your own reverse slash - carriage returns far enough ahead of time to PREEMPT any ones you do not not know about from ever happening.

The usual two methods of preventing unwanted carriage returns are..

(A) Always import text files with carriage returns ADDED AT PARAGRAPH ENDS ONLY.

(B) Always save your work as STANDARD ASCII TEXTFILES, and NOT as word processor documents.

Generally, the newer and the better your word processor or editor, the easier this is to do. Usually it is simply a matter of UNDERSTANDING WHAT IS COMING DOWN.

APPLE IIE

On AppleWriter on the Apple IIe, totally standard ASCII textfiles are used, and there is never any problem inadvertently adding them when importing a text file. Files are saved as standard text files, so there is no problem using them with a comm program.

BUT, if you try printing directly to your PS printer as I do, carriage returns will be forced every 240 characters or less, chopping up longer paragraphs. I automatically solve this problem by using the WPL.200 routine on GENIE PSRT #242 NEWGTOOL.AII. Another route would be to use ProTerm to actually send and receive files.

For more help:
(520) 428-4073

Solving PostScript carriage return hassles...

MACINTOSH

On most Mac word processors, ALWAYS import text files with carriage returns AT PARAGRAPH ENDS ONLY! And ALWAYS use the "SAVE AS" feature, being certain to select a TEXT ONLY output. Then use SendPS to actually send the file to your PostScript printer.

Remember PARAGRAPH ENDS on import; TEXTFILE ONLY on export.

IBM PC OR CLONE

This portion submitted by Richard Kern ...

The MECHANICS GUIDE to GONZO13A

#1 ---- ASCII text and hard returns

In publishing this first book using the gonzo tools and utilities several areas caused an immense amount of grief trying to solve the problem. One of those is the issue of hard returns and getting Gonzo to work around them. Don Lancaster addressed this in the PULP template by inserting \ every 200 or so characters. This told the colcheck proc to ignore the hard returns.

Now I don't like counting characters to find the next 200th one, and I didn't want to write a macro because I found that my word processor was not consistent in the treatment of the files and that would be a waste of time. That question of how the word processor was handling files had to be addressed and solved. The word processor in question is Word Perfect 5.1.

The answer is found in the fundamental nature of word processors. By definition a word processor forces the text into a predefined page definition and will insert softreturns and hardreturns as necessary. However,a text editor does not. The cleanest solution is to use a text editor for all document preparation which involves long strings of text. On a file with "sentences" that are shorter than the screen width, a word processor that will save your file as ASCII text will work fine.

A text editor will let you run characters out on a line as far as you need to, depending on the program. I have several single lines of text that run on for over 600 characters to the right without a hard return and gonzo just loves the stuff.

SPECIFIC PC CARRIAGE RETURN WORKAROUNDS

WordPerfect 5.1 Word Processor

ASCII files can be retrieved but MUST be saved as ASCII text (CNTRL-F5) -1-1. Works fine for program code or small amounts of text. It will insert soft and hard returns.

WP51 has 2 options to import ASCII text: choice #2 on the menu converts all line codes to HRT's and then you are really up the creek. Same comment on usefulness.

Choice #3 on the menu leaves the SRT's but still makes the document conform to its margins and page size. Gonzo get a a nasty cold and acts like a camel - it balks. The output has a commented header and is PS unprintable without modification. So don't use this option.

Workaround is to go into the setup menu and define a new page size. I made one that was 40" wide and now my text strings can run out 40" to the right. Save in ASCII format and you're in PostScript heaven. This is attractive because of the macro capability and command language of WP51.

Don Lancaster's PostScript Secrets

Bonus Supplement #7-B

Carriage return hassles

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #7-C

Carriage return hassles

Solving PostScript carriage return hassles...

Word Perfect Office Editor

Preferred choice, an outstanding product. Allows strings of any length, has macros, search and replace, menus are almost the same as the word processor. The keyboard functions work the same way. No spell check or thesarus (this is a text editor).

Part of the WordPerfect Office package which is normally more $. Even so it's worth the $. It saves you heartache, tears, gnashing of teeth, and other dire consequences to the equipment. I found it by fortuitous accident.

PC Write text editor

Works well and its shareware. It also has macros, search and replace and other features. It will read your ASCII files and write them with out adding any formatting changes. No apparent limit to length of text line! Give it a try and upgrade to the commercial version if you're not a WP fan.

TED.COM - PC Mag text editor

This is your main garden spade for small changes. It loads fast, fast and save your file without forcing its ideas on you. It does have limitations on file size and string width. Don't leave home without it.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

8

Sealing refill holes
Screen Printer mag
Double stroked font
Silk screen suppliers
Diablo page creep fix
Stretching waittimeout

The page creep bug in the Diablo emulation mode on *LaserWriter* Version 38 can be worked around by calling for 65 line pages and then by placing these commands at the start of the very top line of each and every page...

```
[J] [esc] 3 [J] [J] [esc] 4
```

The **[J]** here means "control-J". This is very easy to do using Applewriter's TL command, but may be tricky otherwise. The bug did get fixed in Version 47 and higher. The patch works by removing two pixel lines from each page, which exactly compensates for the page to page creep.

When you are refilling your toner cartridges, a "metallized" write protect tab from a floppy disk makes a quick and dirty tape seal for both the filling hole and the holding tank hole. But a far better solution is to use a nickle plastic *Caplug* from either the *Caplug* people at (716) 876-9855, or from *Niagra Plastics* at (814) 868-3671. Free samples are usually available.

Should you be sending multiple-piece files to your *LaserWriter* and need more than 30 seconds between files for disk loading and processing, just use this command to temporarily extend the default wait timeout...

```
serverdict begin 0 exitserver statusdict /waittimeout 180 put
```

The 180 here is good for 180 seconds, or three minutes. You can change it to anything you like. But avoid ever using the "infinite time" **/waittimeout 0 put** command, especially on a network.

MATERIALS OF THE MONTH –

Two of the leading sources of **SILK SCREEN SUPPLIES** include *Advance Process Supply* at 400 North Noble Avenue, Chicago, IL, 60622; and the *Southern Sign Supply* at 127 Roesler Road, Glen Burnie, MD, 21061. (301) 768-8600.

PUBLICATION OF THE MONTH –

SCREEN PRINTING is a fat and slick monthly magazine serving the silk screen industry. There's lots of new uses and opportunities for your *LaserWriter* output in here. Subscription cost is around $28 per year. *Screen Printing* 407 Gilbert Avenue, Cincinatti, OH, 45202. (513) 421-2050.

FREE FONT

We will be seeing more of double stroked fonts in future columns, but for now, here is a simple outlined and doublestroked font that has adjustable character kerning...

```
/Bookman-Demi findfont [50 0 0 50 0 0] makefont setfont

/workstring (X) def

/doublestrokefont { /textstring exch def /ypos exch def /xpos exch def
textstring stringwidth pop textstring length 1 sub kern mul add 2 div xpos
exch sub ypos moveto textstring {workstring exch 0 exch put workstring
false charpath  gsave 0 setgray fill grestore gsave gsave 0 setgray 1.5
setlinewidth stroke grestore 1 setgray 0.75 setlinewidth currentpoint stroke
grestore exch kern add exch moveto} forall} def

/kern 3 def

200 300 (FREE  FONT) doublestrokefont

showpage quit
```

What you do here is grab each character as an individual text string. You then trace the outline, first stroking wide white, and then narrow black. But note that the characters will fatten excessively if the narrow black gets too wide.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

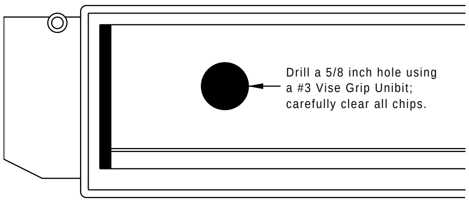
Bonus Supplement #8

Toner Cartridge Refilling Secrets

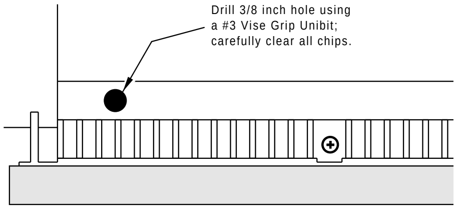
How to refill a toner cartridge...

It is insane not to *personally* refill your own SX and CX toner cartridges, since you can reduce your per page toner costs as much as 15:1 this way. There are two popular methods, called the *punch and go* and the *total teardown*. I very strongly feel that punch and go delivers far and away the lowest end user cost, and that a total teardown often creates more problems than it solves.

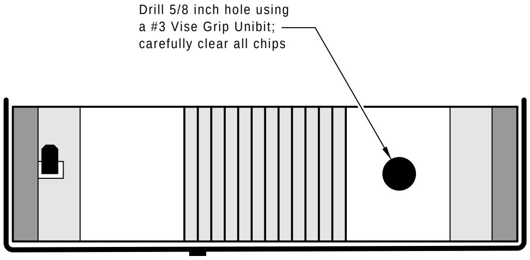
Here's the punch and go details...



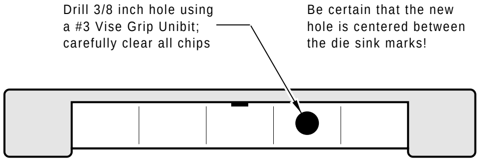
To refill an older CX cartridge with the punch-and-go method, you first snap off the cardboard label and then drill a toner filling hole as is shown here.



A second CX hole is needed to let you empty the spent toner holding tank as shown here. This area is found underneath the cartridge.



The SX cartridge punch-and-go refilling process is similar to the CX, except for the hole locations. The filler hole is shown here.



One or more spent toner drain holes must also get added to the SX cartridge. Since the plastic is thin and brittle, be sure to use a conical step drill.

The peel-and-stick fusion roller wiper pad should also be replaced whenever a cartridge is refilled. Additional details do appear in my *Ask The Guru* reprints, volume II. Two sources of refilling supplies are *Static Control Components* found at (800) 488-2426 and *World Image* at (800) 946-1669.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

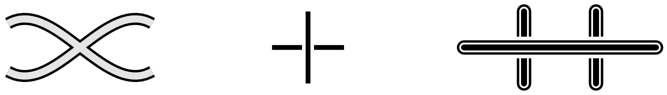
9

It is often easy to lose your place in a very long *PostScript* routine. To ease this, put some obvious and oddball titles on each section, such as...

```
% <<< REAL STUFF STARTS HERE >>>
```

You can do a quick search for STUFF to get to this point in your Postscript code. Standardize on the same unusual section names for all of your work, and you'lll be home free.

Restroking the same line path over again in different shades and widths can create numerous useful effects. For instance, electronic schematic wires can cross over each other if the final line is drawn first as a fat white, followed by a narrow black. Foreground breaks can be similarly added in a tech illustration or an isometric sketch. To draw a pipe, a braid, or a wire, first stroke a fat black, followed by a narrower white or gray...



The **0 setlinecap** gives you flush ends, while a **1 setlinecap** gives you rounded pattern ends, and a **2 setlinecap** gives you ends both boxed and patterned.

To force a *LaserWriter or LaserWriter Plus* reset, switch from 9600 to SPECIAL, wait three seconds, and then switch back. Another method is to send a [C] [D] [C] from your favorite comm program. But note that you cannot switch between AppleTalk and 9600 without a total shutdown and repowering.

MATERIAL OF THE MONTH –

The **#3 VISE GRIP UNIBIT** is ideal for drilling both the filling and emptying holes in the CX or SX toner cartridges. You can reduce your per-page toner costs as much as 15:1 by doing your own refilling, following the secret drilling guides shown you on the previous page. One stocking source of this drill is *Jensen Tools*, 7815 South 46th Street, Phoenix, AZ, 85044. (602) 968-6231.

PUBLICATIONS OF THE MONTH –

There is quite a collection of low cost and **HORRIBLY OUTDATED ART BOOKS** available from *Dover Publications*. In fact, the contents of these books are so far out that they are back in. Art Deco, Thirties stuff, wild typography, unusual border art, and great heaping bunches more. From *Dover Publications*, 31 East Second Street, Mineola, NY, 11501. (516) 294-7000.

FREE FONT

This is Helvetica Bold

This is Helvetica Bolder

Helvetica or most any other type font can be made slightly bolder to match commercial typography simply by repeating it twice side by side. But, this method does degrade the font image and a little bit of stretching goes a very long way. Use a double image only where you absolutely must approximate an existing extra bold type style.

Here's one possible code example...

```
/Helvetica-Bold findfont [28 0 0 28 0 0] makefont setfont

100 100 moveto
/msg (FREE  FONT) def

gsave 3 0 msg ashow grestore 2.5 0 rmoveto 3 0 msg ashow showpage
```

For smaller size fonts, it would be a good idea to combine this with the pixel locking shown in *LaserWriter Secrets #13.*

For more help:
(520) 428-4073

PostScript string de-referencing...

Don Lancaster's PostScript Secrets

Bonus Supplement #9

String de-referencing

Have you ever had a PostScript string inexplicably change its contents or refuse to get recognized halfway through a complex program?

I've had this happen to me a number of times, and it even sometimes shows up in the ADOBE DISTILLERY. The problem is especially common in dictionary stashes. I think I have now found both the cause and the simple but subtle cure, so this short PSRT tech note...

Say you have a string named /workstring...

```
/workstring (ABC) def
```

Lets SUPPOSEDLY save a copy of this string...

```
/oldstring workstring def
```

And /oldstring now, of course contains (ABC). Now the fun begins. Say you change workstring like so...

```
workstring (X) 1 1 putinterval
```

And your workstring now contains (AXC). Now for the trick question: What is now in oldstring?

If you guessed (ABC) you are WRONG!!!. You will find the "NEW" (AXC) in your "old" copy of oldstring!!!!.

Which makes what you thought was in oldstring wrong. Such that it prints wrong, includes extra characters, repeats itself, or else refuses to get properly compared. Causing maddening and infuriating debugging problems.

What to do?

Well, when you attempt an...

```
/oldstring workstring def
```

you do NOT create a new string. All you have done is saved a POINTER that points to the OLD copy of workstring. And if you change workstring with a PUT or a PUTINTERVAL, sure enough, oldstring also gets changed, because oldstring is NOT a string, it is just a pointer right on back to workstring.

There is really JUST ONE string involved.

THE CURE

When you want to save the EXACT contents of some string, you must create NOT A POINTER, but a BRAND NEW STRING that will hold the contents for you.

Like so....

```
workstring dup length string cvs /oldstring exch def
```

This immediately creates a new string and pokes the current values into oldstring. There is no longer any pointer involved, and workstring can go ahead and change all it wants to without changing old string.

We can say that the two strings are DE-REFERENCED. The one has nothing to do with the other any more.

Note that cvs is normally used to change something else into a string. But there is nothing that says you cannot use it to change a string into a new string. Nothing at all.

To rehash: To make sure a string copy does not change later, ALWAYS create a new string by using the cvs operator, rather than just defining a pointer.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

10

Version 47 ROM upgrade kits are now available for your *LaserWriter* or *LaserWriter Plus* through your dealer with a list price around $320. The upgrade runs most everything 1/3 faster, and cures some nasty blowups of version 38, but has a stronger lock on fontpath. Yes, the **framedevice** blowups have been minimized, and those **copypage** hassles have been fixed. The upgrade should rapidly pay for itself if you use your LaserWriter more than two hours per day.

Use this PostScript sequence to do "backwards" printing . . .

```
612 0 translate -1 1 scale
```

You might want to do this for a window decal or other transparency. You can also use **0 796 translate 1 -1 scale** to turn things upside down, or else use a **612 792 translate -1 -1 scale** to work both upside down and backwards.

The baseline setup for your serial communications should be 9600 baud, eight data bits, one stop bit, and no parity. If this doesn't seem to work, try using seven data bits and two stop bits instead. Or call our helpline for assistance.

MATERIAL OF THE MONTH —

The **KROY LAMINATING MATERIAL** is a transparent overlay that is ideal for book covers, menus, and anywhere else heavy handling of a toner image is involved. The material is thinner and much more subtle than your usual peel-and-stick acetates. Available from *Kroy Sign Systems* Scottsdale Airpark, Scottsdale, AZ 85260. (800) 521-4997.

PUBLICATION OF THE MONTH —

A good **TYPOGRAPHY CHEAT BOOK** is super handy for layout and style ideas. My favorite here is Biegleisen's *Art Director's Work Book of Type Faces.* From Arco Publishing, at 219 Park Avenue South, New York, NY, 10003. (212) 777-6300. A few similar titles are available from ads in *U&lc* at (212) 371-0669, or from *North American Publishing* at (215) 238-5300.

True 3-D lettering is almost as easy to do as plain old shadow printing, although it does takes somewhat longer to print.

Here is the code . . .

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

/shadowproc {gsave shadowdepth 2 mul {0 0 moveto spread 0 msg ashow
shadangle cos 2 div shadangle sin 2 div translate} repeat grestore} def

/charproc { gsave 0 0 moveto gsave spread 0 msg ashow grestore 0 setgray
6 setlinewidth stroke grestore} def

/shadowdepth 10 def
/spread 3 def
/shadangle 45 def

% demo -- remove before use --

/AvantGarde-Demi findfont [50 0 0 50 0 0] makefont setfont /msg (FREE
FONT) def

200 300 translate
0.4 setgray shadowproc 0.9 setgray charproc
showpage quit
```

This works by a repeated stagger printing for the shadow, followed by a single printing for the final letter. Even more impressive results can be obtained if you combine this technique with the **charpath** and the **forall** operators. We'll see more on this shortly.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #10

Using those EPS Encapsulated PostScript Files

Using EPS encapsulated PostScript files...

An *Encapsulated PostScript*, or EPS file is some segment of PostScript code that can be safely imported by a larger application program. This lets you move your figures or illustrations into pagemaking programs, or otherwise link any individual or custom routines together.

The two key documents are the *Encapsulated PostScript Files Specification*, Version 2.0, and the *Document Structuring Conventions Specification*, Version 2.1. Both of these are available free from Adobe Systems at (415) 961-4400.

All of the document structuring conventions are voluntary, and are completely ignored by your printer's PostScript interpreter. Thus, cooperation is required between the EPS and host routines to decide what is really needed.

A conforming document comment will begin with a **%%** or a **%!** and will be less than 255 characters long. Extensions can use a **%+** line. Often used comments include the PostScript version, title, creator, and date. Other comments can be included for paper weight, color, included fonts, files, and procs.

On to an EPS file. In its simplest form, an ordinary textfile is used. The only manditory requirements are these two initial lines ...

```
%%!PS-Adobe-2.0 EPSF-2.0
%%BoundingBox: LLx LLy URx URy
```

Here **LLx** is the distance in points from the left sheet edge to the left side of your image. **LLy** is the distance from the bottom sheet edge to the image bottom. **URx** is the distance from the left sheet edge to the right image side and **UR**y is the distance from the bottom sheet edge to the top image limit.

Note that all these numbers are completely independent of any translation, scaling, or rotation system in use within your EPS file. To find your bounding box, you print your file and then measure all these dimensions in points on your printed page. Use a standard point rule to do this. The purpose of a bounding box is to let an applications package know how large the "standard" size of the image is, and where it is to be located from a reference location.

These two conforming comments are the only two actually required by an EPS file. Any additional conforming comments could get added as needed, or as requested by whatever is importing the EPS file. It is suggested that all the conforming comments be ended by a **%%EndComments** final line.

The EPS code has to be well behaved, and must not do anything that would corrupt what the importing applications package is trying to accomplish. The PostScript EPS code should only involve a single page or smaller image. These commands are not allowed...

```
banddevice     copypage       erasepage      exitserver
framedevice    grestoreall    initclip       initgraphics
initmatrix     note           nulldevice     renderbands
```

Several other obvious and device specific operators are also disallowed, such as **initializedisk**. The **setscreen** and **settransfer** operators are allowed only if you carefully do save and restore the original screen and transfer functions.

While the **showpage** operator still is allowed in an EPS file, the importing program is supposed to be intelligent enough to ignore it.

The rules here are simple: *Don't* do anything that messes up the importing program. *Do* make sure the importer can pick up exactly where it left off.

What I have just described is the textfile form of an EPS file. These usually will be host independent. It is also possible to include an EPS file into a device specific host program. This can be done to allow such things as non-PostScript screen images and other host-specific code. Examples of these would include EPSF and PICT screen files on the Mac, and TIFF files on IBM.

Once again, get the original EPS specs before actually using any EPS files.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

11

Sturdy *Macho* font
Auto-collating trick
Rubber stamp glop
Exotic string widths
Curing white stripes
Whole Earth's *Signal*

The *straight through* paper path on your *LaserWriter* NT or NTX prints faster and is less likely to jam. Whenever possible, reverse your page order with software, rather than using the sequence reversing top exit tray. This gets very important on long double sided printing jobs.

If you get a thin and persistent vertical white stripe in your *LaserWriter* or *LaserWriter Plus* copies that does not change when you change cartridges, here is how to fix it – Open the machine and remove the cartridge. Have a friend slide the two tabs above the cartridge hole enough to open the scanning window. Blow into the scanning window by using a solder sucker or else a photographer's air bulb to remove the cathair that is stuck to the mirror. If that does not do it, very carefully and very gently wipe the mirror once with a brand new artist's brush. Note that this is a delicate front surface mirror.

One "sledgehammer" cure to find the width of any text string, no matter how much custom kerning, font changes, or character post-processing is in use – Do a **gsave**, then select the **nulldevice**, and do a **0 0 moveto**. Image the string. Do a **currentpoint**, followed by a **pop**. The stack will now hold the string width.

A somewhat muffled and intermittent yowling from inside your *LaserWriter* can often be cured by opening the lid and letting the cat out.

MATERIAL OF THE MONTH —

MERIGRAPH is an ultra-violet curing liquid resin you can use to convert any Laserwriter printed images into rubber stamps, printing plates, or even 3-D solid objects. Available from *Hercules*, 300 East Shuman, Suite 260, Naperville IL 60566. Their phone number is (800) 323-1832.

PUBLICATION OF THE MONTH —

The new **SIGNAL** book from the Whole Earth folks is crammed cover to cover with outstanding communications tools and resources that span the gamut from desktop publishing to interspecies communications. This "must have" reference is in the tone and spirit of the original Whole Earth Catalogs. From *Whole Earth Review*, 27 Gate Five Road, Sausalito, CA, 94965. (415) 332-1716.

FREE FONT

It's rugged, its macho, and it looks its very best up around 2000 points on a storefront. Yet, whoodathunkit was really Avant Garde in drag? Here's the short and reasonably fast code that is involved...

```
/machofont {save /snap1 exch def /message exch def /fontsize exch def
moveto 1 string /target exch def /AvantGarde-Book findfont [fontsize 0 0
fontsize 0 0] makefont setfont 1 setlinecap 1 setlinejoin /outline fontsize 10
div def message {target exch 0 exch put target false charpath gsave outline
setlinewidth stroke grestore currentpoint exch fontsize 20 div add exch
moveto} forall snap1 restore} def

% demo - remove before use...

/fontsize 80 def /xpos 60 def /ypos 300 def

xpos ypos fontsize (FREE FONT) machofont  showpage quit
```

What is happening here is that you first do a **charpath** for each letter. Then you perform a wide stroking of that path using rounded corners and rounded ends. The corners end up filletted, rather than rounded, since a double stroking is involved. For this font to work, your outline width must completely fill in all of your individual characters.

For more help:
(602) 428-4073

There is a style of envelope called a "French Cut" or "Square Cut" where the flap goes straight across the back, instead of being angled. These often print much better on the LaserWriter. Your local *Xerox* supplies office is one source.

Here's how you put a PostScript integer into a text document . . .

% Put the needed integer on top of the stack. Then do a . . .

(   ) cvs show

This converts the integer into a string and then shows it in the current font at the currentpoint. One very obvious use is for sequential ticket numbering.

The reverse slash is a highly useful "escape" tool which can let you print the unprintable. As some examples, **\r** prints a carriage return, **\261** prints a long dash or **\320** a super-long dash, while a **\360** from the symbol font prints an Apple logo. See the red book for additional codes.

Note that a reverse slash followed by an octal number will print that character. This nicely solves the problem of printing high ASCII codes while you are communicating in strictly 7-bit low ASCII.

MATERIAL OF THE MONTH —

Two extremely useful materials from 3M are their **SCOTCH COLOR KEY** and their **SCOTCHCAL LABEL** product lines. Their Color Key materials include diazo color transparencies intended for color proofing. The Scotchcal stuff easily makes durable vinyl and metal self-stick labels or dialplates in any of five colors and their reverses. From *3-M,* 3-M Center, St. Paul, MN 55144. (612) 733-1110.

PUBLICATION OF THE MONTH —

The **U&lc** is an outstanding typography quarterly with all sorts of typesetting secrets and lots of superb graphic ideas in it. You are supposed to sound like an art director when you ask for your free subscription. From *ITC U&lc* at 2 Dag Hammarskjold New York, NY. 10017 (212) 371-0699.

free font

Some subtle highlights can make a font appear like it is rounded or raised. This effect does take a lot of trial and error and works best on large display messages. Here is one way to do it . . .

```
/currentchar (X) def

/shinyfont{msg {currentchar exch 0 exch put currentpoint /yy
exch def /xx exch def currentchar false charpath clip xx
yy moveto outlinewidth setlinewidth stroke newpath xx
shinywidth lightangle 180 sub cos mul add yy shinywidth
lightangle 180 sub sin mul add moveto currentchar show
initclip newpath xx currentchar stringwidth pop add stretch
add yy moveto} forall} def

% ////  demo - remove before use.  ////

/Bookman-DemiItalic findfont [40 0 0 40 0 0] makefont setfont

/stretch 0 def
/outlinewidth 3 def
/shinywidth 2.5 def
/lightangle 165 def
/msg (free font) def

100 200 moveto msg shinyfont   showpage quit
```

Don Lancaster's PostScript Secrets

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #12-A

Cardboard box layout

Cardboard box layout utility...

This utility and demo shows you how to lay out cardboard packaging boxes and mailers.

Besides total programmability, automatic thickness compensation is included.

```
% uncomment and use these debugging and power tools ONLY when
% GONZO15A.PTL has been persistently downloaded. These tools also VASTLY
% improve the speed and convenience of any related graphics layout...

% gonzo begin
% ps.util.1 begin
% printerror
% nuisance begin


%!PS-Adobe-3.0 EPSF3.0

%%Title:         Layout tools for a type 4001 box
%%Creator:        Don Lancaster and Synergetics (602) 428-4073
%%CreationDate:  May 1, 1993
%%BoundingBox:   Full page; varies with selected size
%%EndComments

100 dict /Boxdict exch def Boxdict begin

systemdict /setstrokeadjust known {true setstrokeadjust} if

% This is the layout for a "type 4001" box. Other boxes will be alike but different
% somehow...

/inches {72 mul} def            % set dimensional units

/length     2     inches def     % the length of the box
/width      1     inches def     % the width of the box
/depth      1.5   inches def     % the depth of the box
/thick      0.10  inches def     % the thickness of the box material
/relief    0.0315 inches def     % total flap relief
/tabdepth  0.250  inches def     % depth of sealing tab
/tabslant  0.200  inches def     % tab slant offset

/xposn      1.0 inches def       % lower lefthand corner x position on page
/yposn      1.5 inches def       % lower lefthand corner y position on page
/scalefactor 1 def              % scale of final drawing
/landscape true def             % portrait (false) or landscape (true)
/showfolds true def             % show the folds?
/showedges true def              % show the edges?
/edgeweight 0.2 def              % weight of edge line in points
/boxgray 0.95 def               % fill color if used

% define some working variables similar to previous non-PostScript code...

/rr {relief 2 div} def                      % half relief thickness
/aa {length thick 2 div add} def             % old A
/bb {width thick add} def                    % old B
/cc {length thick add} def                   % old C
/dd {width thick add} def                    % old D
/ee {tabdepth thick 2 mul add} def           % old E
/ff {depth 2 div thick 2 div add} def        % old F
/gg {depth thick 2 mul add} def              % old G
/hh {depth 2 div thick 2 div add} def        % old H
/ii {tabslant} def                          % old I
/jj {relief} def                            % old J
```

For more help:
(602) 428-4073

Cardboard box layout utility...

```
% pick a non-putrid gray
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen

% /foldproc sets the style of any fold markings...
/foldproc {[1 2] 0 setdash 0.1 setlinewidth stroke} def

% /boxproc sets the style of the box itself...
/boxproc {edgeweight setlinewidth gsave boxgray setgray fill grestore 1
setlinecap stroke} def

% /setpage decides portrait or landscape orientation
/setboxpage {landscape {-90 rotate -792 0 translate} if xposn yposn translate
scalefactor dup scale} def

% /notchup and /notchdown chomp out the side relief pattern...
/notchup {0 ff rr sub rlineto currentpoint exch rr add exch rr 180 0 arcn 0 ff rr sub
neg rlineto} def

/notchdown {0 ff rr sub neg rlineto currentpoint exch rr sub exch rr 0 180 arcn 0
ff rr sub rlineto} def

% /drawbox4001 draws the box. Details will change with other shapes
/drawbox4001 {newpath
            0 0 moveto                    % lower left corner
            aa rr sub 0 rlineto           % first flap
            notchup                       % first notch
            bb jj sub 0 rlineto           % second flap
            notchup                       % second notch
            cc jj sub 0 rlineto           % third flap
            notchup                       % third notch
            dd rr sub 0 rlineto           % fourth flap
            0 ff rlineto                  % right bottom

            tabdepth tabslant rlineto     % end tab
            0 gg tabslant 2 mul sub rlineto
            tabdepth neg tabslant rlineto

            0 hh rlineto                  % right top
            dd rr sub neg 0 rlineto       % fourth flap
            notchdown                     % third notch
            cc jj sub neg 0 rlineto       % third flap
            notchdown                     % second notch
            bb jj sub neg 0 rlineto       % second flap
            notchdown                     % first notch
            aa rr sub neg 0 rlineto       % first flap
            0 ff gg add hh add neg rlineto  % left end

            boxproc                       % and draw it
            } def

% /showfolds4001 draws the bend lines when requested...
/showfold4001 {gsave                          % take snapshot
         0 ff moveto                          % lower edge
         aa bb add cc add dd add 0 rlineto

         0 ff gg add moveto                   % upper edge
         aa bb add cc add dd add 0 rlineto

         aa ff moveto                         % left fold
         0 gg rlineto

         aa bb add ff moveto                  % center fold
         0 gg rlineto

         aa bb add cc add ff moveto           % right fold
         0 gg rlineto

         aa bb add cc add dd add ff moveto    % tab fold
         0 gg rlineto

         foldproc                             % draw it
         grestore } def
```

Don Lancaster's PostScript Secrets

Bonus Supplement #12-B

Cardboard
box layout

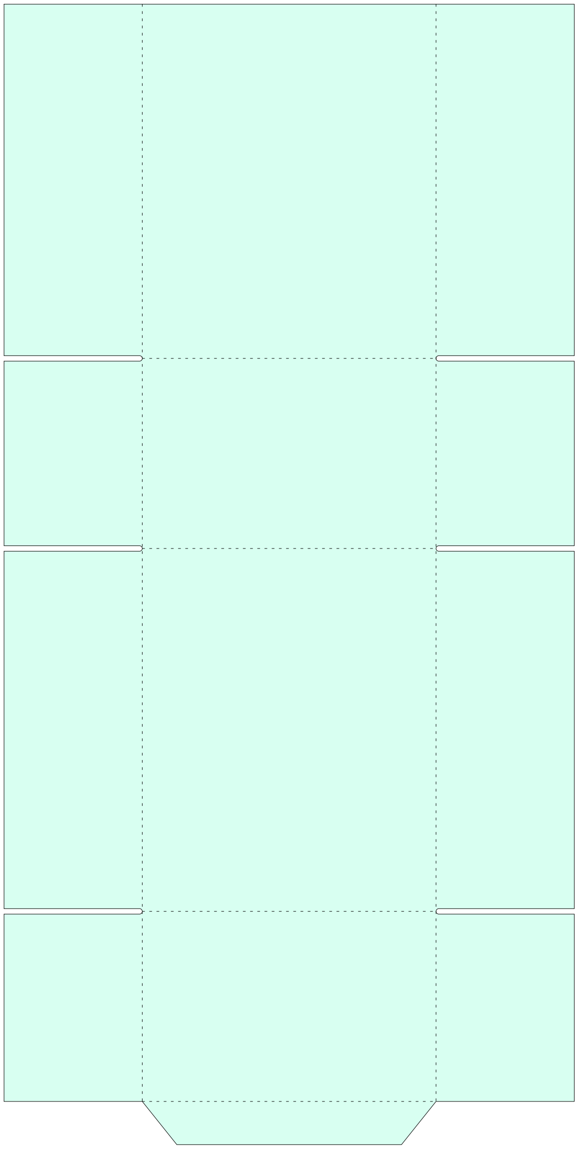
For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #12-C

Cardboard box layout

```
/boxtype4001 {
    save /boxsnap exch def        % take snapshot
    setboxpage                    % position on page
    showedges {drawbox4001} if    % draw the box if requested
    showfolds {showfold4001} if   % draw the folds if requested
    boxsnap restore } def         % and exit

%  //// demo -- remove before reuse ////

% change the variables above or else repeat them here.

boxtype4001              % print the box
showpage                 % and print test copy
end                      % close Boxdict
```



For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

13

Smaller text fonts
Chemical cleaning
Badgemaking stuff
Barcode resources
Electronic Publishing
IIgs AppleTalk hassles

Do **NOT** attempt to use freon or any other chemical sprays when cleaning your *LaserWriter*. Freon converts toner into a conductive and a nearly indestructable varnish. Some electrical cleaners, such as Lectra-Motive #05018, can actually dissolve your *LaserWriter*. Use a solder sucker or a vacuum cleaner instead.

Here's how to get up to date on bar codes. The UPC council has the standards, the AIM is a trade group, while the two others are free trade journals·

A.I.M.
1326 Freeport Road
Pittsburgh, PA 15238
(412) 782-1624

AUTOMATIC ID NEWS
7500 Old Oak Blvd.
Cleveland OH 44130
(216) 243-8100

IDENTIFICATION JOURNAL
2640 N.Halsted Street
Chicago, IL 60614
(312) 528-6600

UPC COUNCIL
7075 Corporate Way S106
Dayton, OH 45459
(513) 435-3870

The *AppleTalk* driver on the IIgs expects and uses all eight data bits as input. This can result in error-generating high-ASCII when getting input from BASIC, *AppleWriter* 2.1, or from other older high-ASCII programs. Using *AppleWriter 2.0* or else a custom machine language driver that clears the high bit resolves this hassle. You could also try using *mark parity* on a newer *LaserWriter*.

MATERIAL OF THE MONTH —

The folks at **BADGE-A-MINIT** offer a complete source of badgemaking tools and supplies that can go as low as eleven cents per badge. From *Badge-A-Minit*, 348 North 30th Road, Box 800 LaSalle, IL 61301. (815) 224-2090.

PUBLICATION OF THE MONTH —

ELECTRONIC PUBLISHING is a free trade journal that has good information on papers, toners, and lots of small quantity desktop publishing tools. They also have a yearly supplier directory. From *Electronic Publishing*, at 401 North Broad Street, Philadelphia, PA 19108. (215) 238-5300.

FREE FONT

The quality of extremely small text can be dramatically improved by using the following routine. You can now print legible text as small as 3 points on your 300 DPI *LaserWriter* printer·

```
/currentchar (X) def

/fineprint {{currentchar exch 0 exch put currentpoint dtransform exch ceiling
exch idtransform moveto currentchar show currentchar ( ) eq {currentpoint
exch spacekern add exch moveto} {currentpoint exch charkern add exch
moveto} ifelse} forall} def

% /// demo - remove before use. ///

/Helvetica findfont [6 0 0 6 0 0] makefont setfont
/pixel {72 mul 300 div} def
/charkern 1 pixel def
/spacekern 1 pixel def

100 200 moveto (FREE FONT) fineprint showpage quit
```

The fineprint code works by always moving at least one whole pixel between successive characters. Note that this routine is intended for portrait only. A separate routine that omits the pair of **exch** commands after **dtransform** can be used for landscape-orientated business cards.

Naturally, you'll get the best results when using plain old *Helvetica* or by going to a simple block lettering custom font.

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

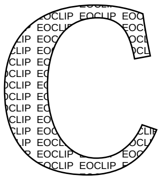
14

Fatter calligraphy
Fast irregular clip
In-Plant magazine
Kerning a number
Kroy Kolor process
Backwards printing

The numbers in practically all fonts are in a constant and fixed pitch, even if the font is proportionally spaced. This is done so that columns of figures will space uniformly. Sometimes, you might like to add negative kerning around the numeral one when it is in text. Thus, (212) 411-7199 does not look nearly as good as (212) 411-7199. One point of negative kerning is used here.

Using an irregular clipping interval can ridiculously slow down your LaserWriter. Instead, put the entire background down first and then do **one** single eoclip to erase everything **outside** the needed area. This method is insanely faster...

```
100 200 translate /Helvetica findfont [7 0 0 7 0 0]
makefont setfont gsave 17 {0 4 moveto 5 {(EOCLIP  )
show} repeat 0 8 translate } repeat grestore
/Bookman-Demi findfont [180 0 0 180 0 0] makefont
setfont 0 0 moveto  150 0 rlineto 0 150 rlineto -150 0
rlineto closepath 10 10 moveto (C) false charpath
eoclip 1 setgray fill initclip 10 10 moveto (C) false
charpath 0 setgray stroke showpage
```



A font can be reversed by changing the sign of the first number in the font matrix. For instance, here is how to do a hand pointing to the left...

```
/ZapfDingbats findfont [-24 0 0 24 0 0] makefont
setfont 100 200 moveto (+) show showpage
```

☜ ☜ ☜

Similarly, you can turn a font upside down by negating the fourth number in the font matrix. Neat, huh?

MATERIAL OF THE MONTH —

KROY-KOLOR is a simple way to convert a black toner image into any of dozens of vibrant colors, using nothing but a second trip through your printer, a special machine, or even an iron. Cost is around fifty cents per sheet and free samples are now available. From *Kroy Kolor*, 14555 North Hayden Road, Scottsdale, AZ 85260 (800) 521-4997.

PUBLICATION OF THE MONTH —

IN-PLANT PRINTER is a free trade journal with lots of good printing ideas in it, along with ads for papers, supplies, and materials which are all suitable for desktop publishing uses. From *In-Plant Printer*, Box 368, Northbrook IL, 60062. The phone number is (312) 564-5940.

Free Font

The Zapf calligraphic font tends to get weak and thin in the larger sizes needed for certificates and awards. Here's how to thicken it (or any other font) by reprinting it several times, each of which is started one pixel to the right of the previous printing...

```
/fatfactor 12 def
/kernfactor 2 def
/showbolder {gsave currentpoint translate /msg exch def  fatfactor cvi {0 0
moveto kernfactor 0 msg ashow 72 300 div 0 translate} repeat grestore} def

/ZapfChancery-MediumItalic findfont [60 0 0 60 0 0] makefont setfont

200 400 moveto
(Free  Font) showbolder showpage quit
```

The fatfactor is the stretch in pixels, while the kernfactor is the extra character spacing in points. These will change with your selection of point size.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #14-A

Font recoding utilities

Adobe font recoding utilities...

Three of the Adobe font recoding modules have been gathered together into one single file for convenience in downloading.

Font recoding lets you add or modify characters in an existing Type I font.

The three modules included are the standard font reencoding algorithm, adding definitions to Type I font CharStrings, and conditionally defining Latin ISO or other composite objects to save VM.

```
%!
% Copyright (C) 1991 by Adobe Systems Incorporated.
% All rights reserved.
%
% This is the standard font reencoding algorithm.
%

/RE { % /NewFontName [NewEncodingArray] /FontName RE -
  findfont dup length dict begin
    { % copy font dictionary into new dict
      1 index /FID ne
      {def} {pop pop} ifelse
    } forall
    /Encoding exch def
    /FontName 1 index def
    currentdict definefont pop
  end
} bind def
```

```
%!
% Copyright (C) 1991 by Adobe Systems Incorporated.
% All rights reserved.
%
% This file is an example of how to add character definitions to
% the CharStrings dictionary of a Type 1 font.
%

/bdef {bind def} bind def

/Max {2 copy lt {exch} if pop} bind def
/Min {2 copy gt {exch} if pop} bind def

/FilledBox { % CharStrings FilledBox -
  pop
  1000 0 100 100 900 900 setcachedevice
  100 100 moveto 900 100 lineto
  900 900 lineto 100 900 lineto
  closepath fill
} bdef

/NotFilledBox { % CharStrings NotFilledBox -
  pop
  1000 0 100 100 900 900 setcachedevice
  100 100 moveto 900 100 lineto
  900 900 lineto 100 900 lineto
  closepath stroke
} bdef

/infinity { % CharStrings infinity -
  pop
  713 0 setcharwidth
  /Symbol findfont 1000 scalefont setfont
  0 0 moveto (¥) show
} bdef
```

For more help:
(520) 428-4073

Adobe font recoding utilities...

```
/RE { % /NewFontName NewEncodingArray FontName RE -
  findfont dup length dict begin
    { % copy font dictionary into new dict
      1 index /FID ne
      {def} {pop pop} ifelse
    } forall
    /Encoding exch def
    /FontName 1 index def

    currentdict definefont pop
  end
} bdef

/REA {   % /NewFontName NewCharsBBox NewEncodingArray
  %   NewCharNamesArray FontName REA -
  % ReEncode font Adding new character procs to CharStrings Dict
  % Does not allow recursive calls to show
  % Not copying UniqueID allows new procs to override old characters
  % Installs a predefined Encoding Vector in new font
  % All new character names must be present in the Encoding Vector
  % Updates the FontBBox to account for BBox of New Characters
  % Application must pre-calculate BBox for New Characters
  % There is ~5KB overhead per font to copy the CharStrings Dict
  % New Character Drawing Procs should be defined in main Driver Dict
  %   this allows the Procs to be shared by many Fonts
  % New Character Procs are responsible for call to setcachedevice
  %   or setcharwidth to specify the widths of the characters

  findfont dup begin
  dup length dict begin
    {       % Copy Top-Level Font Dict
      1 index /FID ne
      2 index /UniqueID ne and
      {def} {pop pop} ifelse
    } forall

    CharStrings dup length   % Size of CharStrings
    2 index length    % Plus Number of New Character Names
    add dict begin    % Create and Add to Dict Stack
      {def} forall   % Copy existing CharStrings entries
      {     % Define new Procedures in CharStrings
        dup load def
      } forall
      currentdict
    end
    /CharStrings exch def  % Define new CharStrings Dict
    /Encoding exch def  % Define new Encoding Array

    aload pop    % Put BBox values for new Chars on stack
    3 -1 2 {    % Calculate new urx and ury values
      /FontBBox load
      exch get Max   % Leave larger of two values on stack
      4 1 roll
    } for
    1 -1 0 {    % Calculate new llx and lly values
      /FontBBox load
      exch get Min   % Leave smaller of two values on stack
      4 1 roll
    } for
    4 array astore    % Create and define new BBox array
    /FontBBox exch def
    /FontName 1 index def
    currentdict definefont pop
  end
  end
} bdef
```

Don Lancaster's PostScript Secrets

Bonus Supplement #14-B

Font recoding utilities

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #14-C

Font recoding utilities

Adobe font recoding utilities...

```
/Times-RomanISO ISOLatin1Encoding /Times-Roman RE
/Times-RomanISO findfont 12 scalefont setfont
100 200 moveto
(Times-RomanISO) show

/MyEncoding
  StandardEncoding 256 array copy
  dup 0 /FilledBox put
  dup 1 /NotFilledBox put
  dup 65 /NotFilledBox put
  dup 165 /infinity put
def
/MyCharNames [
  /FilledBox
  /NotFilledBox
  /infinity
] def
/MyCharsBBox [25 100 900 900] def
/MyTimes-Roman MyCharsBBox MyEncoding MyCharNames /Times-Roman REA

/MyTimes-Roman findfont 12 scalefont setfont
100 100 moveto
(A ¥) show

showpage
```

```
%!
% Copyright (C) 1991 by Adobe Systems Incorporated.
% All rights reserved.
%
% filename      : iso.ps
% date created  : 24-feb-91
% last updated  :  6-aug-91
% author        : ross a jeynes
% purpose       : Conditionally define the ISO Latin1 Encoding vector, without
%                 using VM if the vector already exists.  This saves about 4K of VM.
%
%                 This conditional definition technique can be applied to other
%                     types of composite objects as well.

/tmpvm vmstatus pop exch pop def

/SkipISO? /ISOLatin1Encoding where { pop true }{ false } ifelse def
SkipISO? { /dontload save def } if

/ISOLatin1Encoding
[/.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef
/.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef
/.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef
/.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef
/space /exclam /quotedbl /numbersign /dollar /percent /ampersand
/quoteright /parenleft /parenright /asterisk /plus /comma /minus /period
/slash /zero /one /two /three /four /five /six /seven /eight /nine
/colon /semicolon /less /equal /greater /question /at /A /B /C /D /E /F
/G /H /I /J /K /L /M /N /O /P /Q /R /S /T /U /V /W /X /Y /Z /bracketleft
/backslash /bracketright /asciicircum /underscore /quoteleft /a /b /c /d
/e /f /g /h /i /j /k /l /m /n /o /p /q /r /s /t /u /v /w /x /y /z
/braceleft /bar /braceright /asciitilde /.notdef /.notdef /.notdef
/.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /.notdef
/.notdef /.notdef /.notdef /.notdef /.notdef /.notdef /dotlessi /grave
/acute /circumflex /tilde /macron /breve /dotaccent /dieresis /.notdef
/ring /cedilla /.notdef /hungarumlaut /ogonek /caron /space /exclamdown
/cent /sterling /currency /yen /brokenbar /section /dieresis /copyright
/ordfeminine /guillemotleft /logicalnot /hyphen /registered /macron
/degree /plusminus /twosuperior /threesuperior /acute /mu /paragraph
```

For more help:
(520) 428-4073

Adobe font recoding utilities...

```
/periodcentered /cedilla /onesuperior /ordmasculine /guillemotright
/onequarter /onehalf /threequarters /questiondown /Agrave /Aacute
/Acircumflex /Atilde /Adieresis /Aring /AE /Ccedilla /Egrave /Eacute
/Ecircumflex /Edieresis /Igrave /Iacute /Icircumflex /Idieresis /Eth
/Ntilde /Ograve /Oacute /Ocircumflex /Otilde /Odieresis /multiply /Oslash
/Ugrave /Uacute /Ucircumflex /Udieresis /Yacute /Thorn /germandbls
/agrave /aacute /acircumflex /atilde /adieresis /aring /ae /ccedilla
/egrave /eacute /ecircumflex /edieresis /igrave /iacute /icircumflex
/idieresis /eth /ntilde /ograve /oacute /ocircumflex /otilde /odieresis
/divide /oslash /ugrave /uacute /ucircumflex /udieresis /yacute /thorn
/ydieresis ]  def

SkipISO? { dontload restore } if

pstack

vmstatus pop exch pop tmpvm sub pstack
```

Don Lancaster's
PostScript
Secrets

Bonus Supplement #14-D

Font recoding utilities

For more help:
(520) 428-4073

Remember that any unbound PostScript procedure can be redefined at any time for any reason. For instance, **/showpage** can be redefined to put a border and three-hole punch marks on each sheet, or to automatically put down your letterhead and signature around the body of a letter, or to add a large "proof copy" across an artwork image.

You can also switch between "old" and "new" definitions by creative use of new dictionaries. The rule is that the *top* dictionary on the dictionary stack is scanned first for procedural commands.

A font character is stored two different ways in PostScript. It is held as an integer from 0 to 255 by such PostScript operators as **get**, **put**, **read**, **readline**, and **readstring**. It is held as an actual string by **show**, **widthshow**, **awidthshow**, **stringwidth**, **charpath**, and **getinterval**.

```
% getting FROM 0-255 TO string: put integer 0-255 on stack. Then ...
   /str (X) def str exch 0 exch put str
% the string character is now on the stack

% getting FROM string TO 0-255: put string character on stack. Then ...
   /str exch def str 0 get
% the integer 0-255 is now on the stack.
```

MATERIAL OF THE MONTH —

FORM-X FILM is a self-adhesive vinyl film on a carrier that's available in dozens of gloss and matte, transparent and opaque colors. It laser prints beautifully. By printing outlines with your LaserWriter and then cutting those outlines with an X-acto knife, you can eliminate the need for a $10,000 signmaking machine. The same #8 catalog that describes FORM-X also is an excellent "cheat book" for borders and font typography. Available in most larger art stores. From the *Graphic Products Corporation*, Rolling Meadows, IL, 60008. (312) 537-9300

PUBLICATION OF THE MONTH —

SYSTEMDICT is the official monthly news publication of the Adobe Developer's Group, and is full of inside info on the newest and best PostScript info. From *Adobe Systems*, 1585 Charleston Road, Box 7900 Mountain View, CA 94039. The phone number is (415) 961-4400.

free font

Sometimes a few minor changes can dramatically improve on an existing font. Here we have taken Bookman Demi Italic and have fattened it a tad and then de-harshened it by a slight rounding. It was then filled in with a secret dense gray. A fat white pre-stroking creates automatic baseline breaks, a technique very useful for letterheads. Like so . . .

```
/outlineshow { gsave /msg1 exch def translate 0 0 moveto 1 setlinecap 1
setlinejoin msg1 {str exch 0 exch put gsave str false charpath gsave 4
setlinewidth 1 setgray stroke grestore gsave 0.99 setgray fill grestore 1
setlinewidth stroke grestore currentpoint exch str stringwidth pop add kern
add exch moveto} forall grestore } def

/str (X) def  /kern 1 def

%  //// demo - remove before use. ////

/Bookman-DemiItalic findfont [50 0 0 50 0 0] makefont setfont
135 25 {dup mul exch dup mul add 1.0 exch sub} setscreen

200 400 moveto -20 -5 rmoveto 200 0 rlineto 1 setlinecap gsave 4.5
setlinewidth stroke grestore gsave 0.99 setgray 3 setlinewidth stroke
grestore newpath

200 400 (free  font) outlineshow  showpage quit
```

Don Lancaster's PostScript Secrets

15

For more help:
(602) 428-4073

There is a serious communications bug in the LaserWriter NTX. If you set the serial baud rate to a higher value, it will stay there only until you try to use AppleTalk. Use of AppleTalk resets all the serial ports back to default values. Yes, even on power down. Solutions include not ever using AppleTalk at all, or reprogramming the ports every time you do, or creative use of the new and unique **sethardwareiomode** command. Arrgh.

PostScript's **-save-** and **-restore-** commands are not perfect. Any strings you modified with **-put-** will stay modified after a **-restore-**. While really a bug, you can use this "crack" to pass variables, messages, or whatever beyond a saved context. Adobe is violently opposed to you doing this.

The "flying wedge" is a layout trick that dates back to the linotype days. It is used to let random text exactly fill a vertical area. The white space you see above and below all the horizontal gray bars is a PostScript variable that will automatically stretch or compress the text to fit exactly one **Computer Shopper** column. Sneaky, eh what? The code appears on the next page.

MATERIAL OF THE MONTH —

PADDING COMPOUND is some cheap glop that comes in a jar. You use it for forms, invoices, prescription and note pads, calenders, and anywhere else you want a light use or temporary binding. Normally available in white and red, you can add food coloring or other dye to get any color. One source is *Paper Plus*, 3820 South Palo Verde #108, Tucson, AZ 87514. (520) 889-9500.

PUBLICATION OF THE MONTH —

THE GREEN BOOK, otherwise known as **PostScript Language Program Design**, is the third book in the Adobe series on all the basics of PostScript programming. Topics include emulators, page layouts, error handling, scanning, debugging, and file handling. While both handy and useful, it simply is not in the same league as the essential red and blue books. Published by *Addison Wesley*. One stocking source is *Synergetics*, Box 809, Thatcher, AZ, 85552. (520) 428-4073.

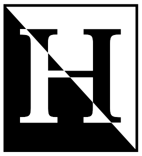
FREE FONT

The **-charpath-** and the **-eofill-** PostScript font operators can often be combined to get some interesting and useful "reversal" effects . . .

```
/str (X) def

/mafiafont {{str exch 0 exch put str stringwidth pop spread add inset add
/xwidth exch def spread neg baseheight neg rmoveto 0 cementheight rlineto
xwidth 0 rlineto 0 cementheight neg rlineto xwidth neg 0 rlineto spread
baseheight rmoveto str false charpath eoclip currentpoint fill newpath exch
spread add exch moveto initclip} forall} def

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

/AvantGarde-Demi findfont [50 0 0 50 0 0] makefont setfont

/spread 2 def /inset 5 def /cementheight 28 def /baseheight
10 def 100 300 moveto ( FREE  FONT ) mafiafont showpage quit
```

And here's how to handle a single split letter . . .

```
/Palatino-Bold findfont [100 0 0 100 0 0] makefont
setfont 100 400 moveto gsave 100 0 rlineto 0 110
rlineto -100 0 rlineto closepath 2 setlinewidth stroke
grestore 0 110 rlineto 100 -110 rlineto  -100 0 rlineto 8
21 rlineto (H) false charpath eoclip fill showpage quit
```



Don Lancaster's PostScript Secrets

16

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #16

Using the Flying Wedge

Using the Flying Wedge...

The flying wedge is a tool that lets you automaticaly stretch the vertical spaces between paragraphs, art cuts, or whatever so they exactly fill up a needed area. The ultimate details will very much depend upon your selection of justification routines and your programming style.

For instance, let's suppose you wanted to space each paragraph with a group of three bullets (● ● ●). We might start off with these PostScript procs lifted from my *gonzo justify* routines...

```
ypos         % is the current text vertical position
xpos         % is the current text left margin
txtwide      % is the width of the text line
cc           % is your center justification routine
fixedwedge   % is the fixed spacing above or below the wedge
wedgeadjust  % is the flexible spacing above or below the wedge
```

Then we might define a **wedgeproc** that creates the actual wedge artwork, and a calling of **flyingwedge** that stretches down, calls **wedgeproc**, and stretches down some more...

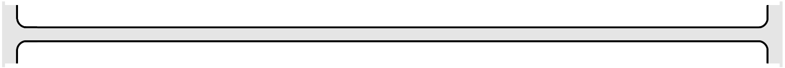
```
/wedgeproc {xpos txtwide 2 div add (\267  \267  \267) cc} def

/flyingwedge {/ypos ypos fixedwedge sub wedgeadjust sub wedgeproc /ypos
ypos gixedwedge sub wedgeadjust sub} def
```

Your **flyingwedge** can be activated by a suitable macro. I define **fmacro** to do an automatic flying wedge on any text insertion of an **[esc]-f**.

The amount of **wedgeadjust** to use is found by dividing the amount of space to be stretched or squashed by *twice* the number of wedges being used. This can be done manually or automatically, although auto-wedging can get downright dangerous at times.

More on the gonzo justify appears in *Ask The Guru*, volume II, and over in my *PostScript Beginner Stuff*.

Whoops, there went one now. Instead of bullets in these *LaserWriter Secrets*, I use a rail. It looks something like this, only a tad larger...



Here's another real one ...

But the flying wedge won't help you if you can't think of anything else to say.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

17

Font and Function
Self-Adhesive films
Sight-reading eexec
A clip and stroke font
Imagewriter emulation
2-way communications

It is extremely important to make sure your *LaserWriter* is set up for two-way communications, so you can receive and optionally record all of your return error messages . . .

On the **MAC**, use **SEND.PS** from AppleTalk or else use **FREETERM** or **RED RYDER** from the faster serial port.

On the **IBM**, use the serial **COM1** port and suitable two-way comm software, such as **CROSSTALK** or **PROCOMM**. Do **not** just copy to COM-1!

On the **Apple**, use **AppleWriter** and its built-in modem **[Q]-I** feature, or else use **PROTERM** or other comm software.

There is an Apple IIgs file called **/SYSTEM.MASTER/APPLETALK /IWEM**. This is an imagewriter emulator which is an ordinary textfile that can be transferred and used as is with any other personal computer. You can also easily rewrite **IWEM** to give you the equivalent of **SEND.PS** on the IIgs. Be sure to define **_WBJ_** as a null proc when you do this. **IWEM** is easily customized any way you like.

Here's a quick summary of how to sight read an **eexec** file: Get into a two-way communication environment that lets you receive and record error messages. Use an error trapper that dumps the stack. Use a **chopper** tool to send an eexec file only up to a cursed character, followed by a [d]. Or, use a **inserter** tool that adds an extra FF or AA into the eexec data stream. The **eexec** file can then be reconstructed from the received and recorded return error messages.

MATERIAL OF THE MONTH —

FSK SELF-ADHESIVE PRODUCTS are covered in a free sample swatch book that includes clear self-stick mylar, silvers, golds, and vinyls. Several of these are laser printable, while others are ideal for the protective overlays for menus, booklets, or whatever. Cost is around ten cents per sheet. From *International Films Inc.*, 45 Industrial Parkway, Woburn MA, 01888. (800) 633-0140.

PUBLICATION OF THE MONTH —

FONT & FUNCTION is a new Adobe PostScript typeface sample book and catalog that also includes lots of typography secrets and layout tips. Free from *Adobe Systems Inc.*, 1585 Charleston Road, Box 7900, Mountain View, CA, 94039. The phone number is (800) 833-6687.

free font

A true outline clipping and a multiple stroking can be combined to produce some "old fashioned" lettering effects that look particularly good on very large display letters . . .

```
/stretch 1  def

/str (X) def

/doubleoutline {gsave { initclip str exch 0 exch put str print flush str false
charpath gsave clip gsave fill grestore gsave 2.5 setlinewidth 1 setgray
stroke grestore gsave 0.5 setlinewidth stroke grestore grestore currentpoint
exch stretch add exch newpath moveto} forall grestore } def

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

/AvantGarde-DemiOblique findfont [80 0 0 70 0 0] makefont setfont
100 200 moveto (FREE FONT) doubleoutline showpage quit
```

This works best with larger and bolder fonts, and you do have to watch out for miter problems. You could repeat the black/white stroking process a few more times can give various "neon tube" effects.

For more help:
(602) 428-4073

Don Lancaster's LaserWriter Secrets

Bonus Supplement #17

Writing and reading *eexec* text files

Reading and writing your own eexec files . . .

Here's how to create your own **eexec** file...

```
/mask 16#D971 def /mult1 16#6000 def /mult2 16#6E6D def
/adder 16#58BF def /strrx (X) def /char 32 def
/hexvalues (0123456789ABCDEF) def

/printashex {cvi /vall exch def vall 16 div cvi hexvalues exch 1 getinterval
print vall 15 and hexvalues exch 1 getinterval print flush 20 {37 sin pop}
repeat formatcount 1 add 32 eq {(\n) print flush 100 {37 sin pop} repeat} if
/formatcount formatcount 1 add cvi 31 and def} def

/makeeexecfile {/formatcount 0 def 4 {char mask -8 bitshift or char mask -8
bitshift and not and /echar exch def echar printashex flush 15 {37 sin pop}
repeat mask echar add dup mult1 mul 16#FFFF and exch mult2 mul 16#FFFF
and add 16#FFFF and adder add 16#FFFF and /mask exch def} repeat
{currentfile strrx readstring {0 get /char exch def char mask -8 bitshift or char
mask -8 bitshift and not and /echar exch def echar printashex flush 15 {37 sin
pop} repeat mask echar add dup mult1 mul 16#FFFF and  exch mult2 mul
16#FFFF and add 16#FFFF and adder add 16#FFFF and /mask exch def} {pop
exit} ifelse} loop} def

% //// demo - remove before use. ////

1500 {37 sin pop} repeat

% Here is the expected host-returned blackflashing result ...

% F983EF00C334F148421509DC30FA053D6DD4273E416E6A2EA64F917B5D20E111
% 9F220AF8FC50D545AB51A0D18B6DD7543D27A21CD55887C1C7D51608F6A316EE
% 8891D92A6E0D09D1D039159DA3A0781E1380B1228C

makeeexecfile
0 0 moveto 1000 0 rlineto 0 1000 rlineto -1000 0 rlineto closepath fill
showpage
```

Note that your **eexec** file gets returned to the host for recording.

And here's how to verify your newly created **eexec** file...

```
% This tests the sample file created by the makeeexecfile demo,
% and should eject one blackflashed page.

currentfile eexec
F983EF00C334F148421509DC30FA053D6DD4273E416E6A2EA64F917B5D20E111
9F220AF8FC50D545AB51A0D18B6DD7543D27A21CD55887C1C7D51608F6A316EE
8891D92A6E0D09D1D039159DA3A0781E1380B1228C
```

Finally, here's how to read an existing **eexec** file...

```
/mask 16#D971 def /mult1 16#6000 def /mult2 16#6E6D def
/adder 16#58BF def /strrx (X) def /skip4 -4 def

/readeexecfile {{currentfile strrx readhexstring{0 get /echar exch def echar
mask -8 bitshift or echar mask -8 bitshift and not and /char exch def skip4 0
ge {strrx 0 char put strrx print flush 15 {37 sin pop} repeat /skip4 skip4 1
add def}{/skip4 skip4 1 add def} ifelse mask echar add dup mult1 mul
16#FFFF and  exch mult2 mul 16#FFFF and add 16#FFFF and adder add
16#FFFF and /mask exch def}{pop exit} ifelse} loop} def

% //// demo - remove before use. ////

1500 {37 sin pop} repeat

% Here is the expected host-returned result for this demo . . .

% 0 0 moveto 1000 0 rlineto 0 1000 rlineto -1000 0
% rlineto closepath fill showpage

readeexecfile
F983EF00C334F148421509DC30FA053D6DD4273E416E6A2EA64F917B5D20E111
9F220AF8FC50D545AB51A0D18B6DD7543D27A21CD55887C1C7D51608F6A316EE
8891D92A6E0D09D1D039159DA3A0781E1380B1228C
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

18

Paper Plus stores
Unusual boxit font
Uhlrichts Dictonary
Printing to the edge
Reverse slash trickery
New LaserWriter grays

For some strange reason, most people and most applications insist on using the seventeenth most putrid LaserWriter gray. Use these instead...

The overall "best" gray –
```
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen
```

A good "india ink wash" gray –
```
133 25 {dup mul exch dup mul add 1.0 exch sub} setscreen
```

One possible "reduced repro" gray –
```
85 35 {dup mul exch dup mul add 1.0 exch sub} setscreen
```

Both the Laserwriter NT and NTX will print nearly to the edges of a full legal sheet size. Older LaserWriters were limited to a smaller legal image area. Some new legal documents may not image properly on the older machines.

Some PostScript code listings will not print properly if you dump them to an emulator or page maker. Note that most reverse slashes should be replaced with three slashes and a space. Reverse slashes that force a line-ending return should be replaced with three slashes. Thus...

\\\ 033 will print as \033

Embedded control characters should be replaced by their octal equivalents.

MATERIAL OF THE MONTH —

The nationwide **PAPER PLUS** chain is where you go for good prices on labels, bumperstickers, parchment, letterheads, astrobrites, and bunches more...

(203) 724-9671
(209) 445-0911
(214) 340-2596
(301) 499-1121
(305) 776-4529
(414) 781-2882
(503) 345-3223
(503) 238-3607
(505) 881-8484
(509) 525-0229
(602) 889-9500
(602) 937-2703
(609) 234-0110
(614) 846-7030
(616) 929-1221
(707) 542-7012
(713) 785-1292
(714) 369-0470
(804) 855-0082
(805) 831-8646

(312) 922-4015
(313) 930-1600
(315) 425-1544
(316) 265-2475
(407) 687-0331
(512) 820-3462
(512) 454-8741
(516) 248-3031
(518) 426-5555
(602) 834-1815
(619) 562-0811
(619) 292-5608
(702) 731-2158
(703) 684-0066
(704) 521-8040
(813) 884-2893
(817) 633-3959
(818) 247-0620
(904) 398-8586
(919) 272-4118

PUBLICATION OF THE MONTH —

That **UHLRICHT'S PERIODICALS DICTIONARY** on the reference shelf of your local library holds the keys to the kingdom. It lists many tens of thousands of free trade journals. Included are printing, signmaking, art, and typography.

FREE FONT

The **forall** procedure can be used to put most any border or special effect around any existing font characters. The boxes automatically get wider for "W" and narrower for "I", and spaces are bypassed. Like so...

```
/belowbase 8 def /leftofchar 5 def /abovebase 36 def
/rightofchar 5 def /tweenchars 18 def /str (X) def

/boxitproc { /msg exch def save /bsnap exch def xpos ypos translate msg {
/char exch def str 0 char put str ( ) ne {gsave currentpoint  belowbase sub
exch leftofchar sub exch moveto 0 belowbase abovebase add rlineto
leftofchar str stringwidth pop add rightofchar add 0 rlineto 0 belowbase
abovebase add neg rlineto closepath gsave fill grestore gsave clip  gsave 3
setlinewidth 1 setgray stroke grestore 1 setlinewidth 0 setgray stroke
grestore newpath grestore } if 1 setgray str show  0 setgray currentpoint
exch tweenchars add exch moveto} forall clear bsnap restore } def

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

/Helvetica-Narrow-Bold findfont [40 0 0 40 0 0] makefont setfont
100 200 moveto (FREE FONT) boxitproc showpage quit
```

For more help:
(520) 428-4073

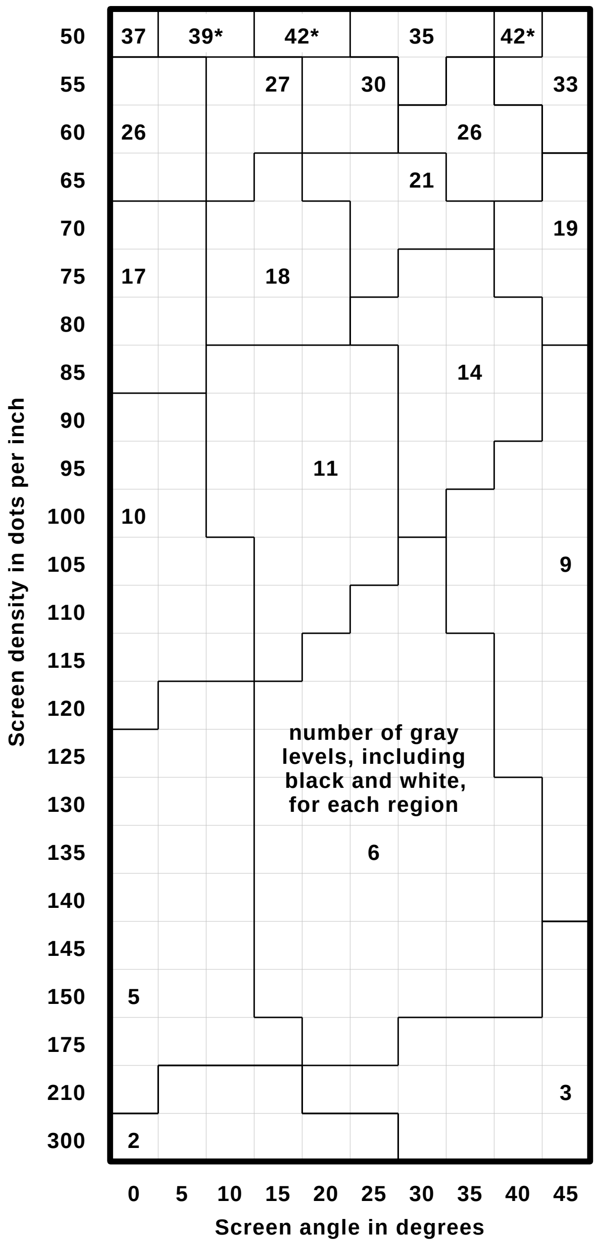
Don Lancaster's PostScript Secrets

Bonus Supplement #18

The Infamous
Secret 300 DPI
Gray Map

The secret 300 DPI gray map . . .

Most users and most applications packages do seem to insist on using the *seventeenth* lousiest gray available on your *LaserWriter. Here's the secret map from my Ask the Guru II,* to all of the other (and usually better) grays . . .



Because of the integer "tile" arithmetic that is involved, any other values will get converted to one of the numbers shown in the above map. The "best" gray is often a **106** line, **45** degree, **9** level gray, set with this command . . .

```
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen
```

Other interesting grays are a **85 35** for reduced reproduction, **75 15** for halftone photos, and **135 25** for *india ink wash* effects. The putrid default gray is **53 45**.

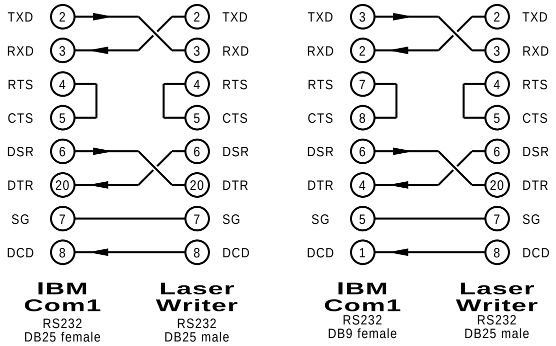
For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

19

Pixie Dust magic
Creating an array
Arc-justified fonts
Forms Professional
Secrets of superexec
IBM/LaserWriter cables

Here's the proper cables to get from an IBM 25 pin or 9 pin COM-1 port to your LaserWriter serial input . . .



For complete return error messages or when compiling routines, use *CrossTalk* or a similar comm program, rather than attempting a simple COM-1 copy.

Be very careful how you create and use an array. If you use the stack to create or **aload** your array, you are limited to a maximum of 500 or so array values. If you create your array ahead of time and use **put** and **get**, then you are allowed as many as 65536 array values.

Here's how to activate **superexec**: First, you do a **1183615869 internaldict begin**. From that point on, a **{forall} superexec** or a **{get} superexec** will often override and ignore any possible **invalidaccess** errors. This greatly expands your abilities to customize PostScript code and operations.

MATERIAL OF THE MONTH —

PIXIE DUST is a $3 per packet powder that you can dust onto the drum of your LaserWriter, LaserWriter Plus, NT, or NTX. It can significantly reduce the number and the severity of the drum scratches. Do a very light dusting whenever you install or refill your cartridges. From *Lazer Products*, 12741 East Caley Avenue, Englewood, CO 80111. (303) 792 5277.

PUBLICATIONS OF THE MONTH —

If you are at all into any forms or form building, check into the free **FORMS PROFESSIONAL** and that **BUSINESS FORMS AND SYSTEMS**. From North American Publishing, 401 North Broad Street, Philadelphia, PA 19108. (215) 238-5300.

FREE FONT

Here's an arcjustify routine that is far faster, simpler, and much easier to kern than the pair in the blue book . . .

```
/arckern 2 def   /str ( ) def

/arcjustify {gsave /msg exch def /radius exch def translate msg dup
stringwidth pop exch length 1 sub arckern mul add 2 div dup 57.29578 mul
radius div msg {str exch 0 exch put gsave rotate 0 radius moveto str dup
dup stringwidth pop 2 div 57.29578 mul radius div neg rotate show
stringwidth pop arckern add sub dup 57.29578 mul radius div grestore}
forall pop pop grestore} def

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

/Palatino-Roman findfont [100 0 0 100 0 0] makefont
setfont 200 0 500 (FREE  FONT) arcjustify   showpage
```

For upward curving text, use a *positive* radius. For downward curving text, use a *negative* radius. Note that **arckern** sets the between-character spacing.

For more help:
(602) 428-4073

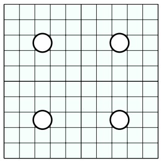
Don Lancaster's LaserWriter Secrets

Bonus Supplement #19

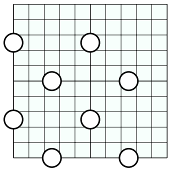
Actual 300 DPI Halftone Spot Patterns

300 DPI Halftone Spot Patterns...

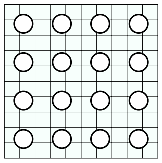
Here are the actual halftone spot pattern locations as used in the LaserWriters and most other 300 DPI PostScript printers...



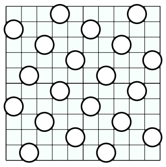
300 SPI, 0 degrees
2 grays



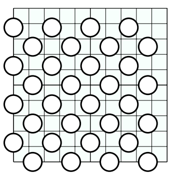
212 SPI, 45 degrees
3 grays



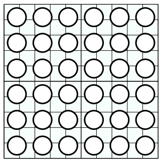
150 SPI, 0 degrees
5 grays



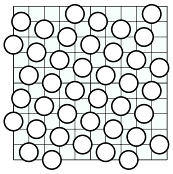
134 SPI, 27 degrees
6 grays



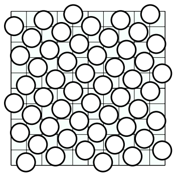
106 SPI, 45 degrees
9 grays



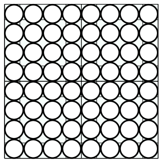
100 SPI, 0 degrees
10 grays



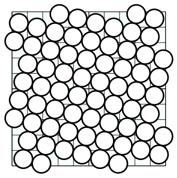
95 SPI, 18 degrees
11 grays



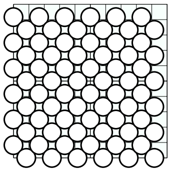
83 SPI, 33 degrees
14 grays



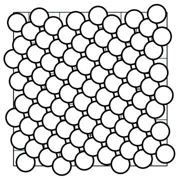
75 SPI, 0 degrees
17 grays



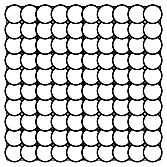
73 SPI, 14 degrees
18 grays



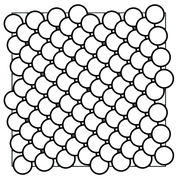
71 SPI, 45 degrees
19 grays



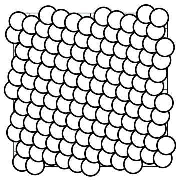
67 SPI, 27 degrees
21 grays



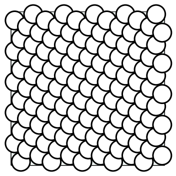
60 SPI, 0 degrees
26 grays



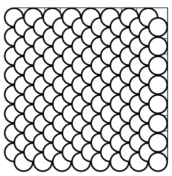
60 SPI, 37 degrees
26 grays



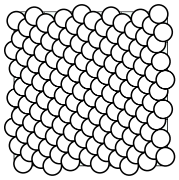
59 SPI, 15 degrees
27 grays



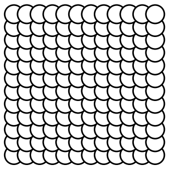
56 SPI, 23 degrees
30 grays



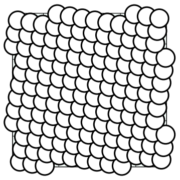
53 SPI, 45 degrees
33 grays



50 SPI, 31 degrees
35 grays



50 SPI, 0 degrees
37 grays



49 SPI, 10 degrees
39 grays

The key secret to generating these spots is that the *setscreen* operator places each spot *x* and *y* coordinate on the stack in turn. Your custom "spotfunction" can then use these pairs any way you like.

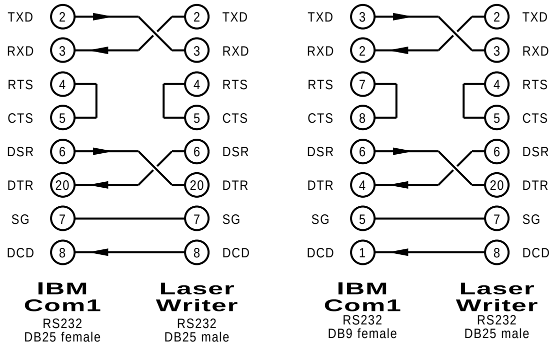
Instead of actually generating a spotfunction, you instead draw graphs or return key information to your host for recording.

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

20

Here's the proper cables to get from an IBM 25 pin or 9 pin COM-1 port to your LaserWriter serial input . . .



For complete return error messages or when compiling routines, use *CrossTalk* or a similar comm program, rather than attempting a simple COM-1 copy.

Be very careful how you create and use an array. If you use the stack to create or **aload** your array, you are limited to a maximum of 500 or so array values. If you create your array ahead of time and use **put** and **get**, then you are allowed as many as 65536 array values.

Here's how to activate **superexec**: First, you do a **1183615869 internaldict begin**. From that point on, a **{forall} superexec** or a **{get} superexec** will often override and ignore any possible **invalidaccess** errors. This greatly expands your abilities to customize PostScript code and operations.

MATERIAL OF THE MONTH —

PIXIE DUST is a $3 per packet powder that you can dust onto the drum of your LaserWriter, LaserWriter Plus, NT, or NTX. It can significantly reduce the number and the severity of the drum scratches. Do a very light dusting whenever you install or refill your cartridges. From *Lazer Products*, 12741 East Caley Avenue, Englewood, CO 80111. (303) 792 5277.

PUBLICATIONS OF THE MONTH —

If you are at all into any forms or form building, check into the free **FORMS PROFESSIONAL** and that **BUSINESS FORMS AND SYSTEMS**. From North American Publishing, 401 North Broad Street, Philadelphia, PA 19108. (215) 238-5300.

FREE FONT

Here's an arcjustify routine that is far faster, simpler, and much easier to kern than the pair in the blue book . . .

```
/arckern 2 def   /str ( ) def

/arcjustify {gsave /msg exch def /radius exch def translate msg dup
stringwidth pop exch length 1 sub arckern mul add 2 div dup 57.29578 mul
radius div msg {str exch 0 exch put gsave rotate 0 radius moveto str dup
dup stringwidth pop 2 div 57.29578 mul radius div neg rotate show
stringwidth pop arckern add sub dup 57.29578 mul radius div grestore}
forall pop pop grestore} def

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

/Palatino-Roman findfont [100 0 0 100 0 0] makefont
setfont 200 0 500 (FREE  FONT) arcjustify   showpage
```

For upward curving text, use a *positive* radius. For downward curving text, use a *negative* radius. Note that **arckern** sets the between-character spacing.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

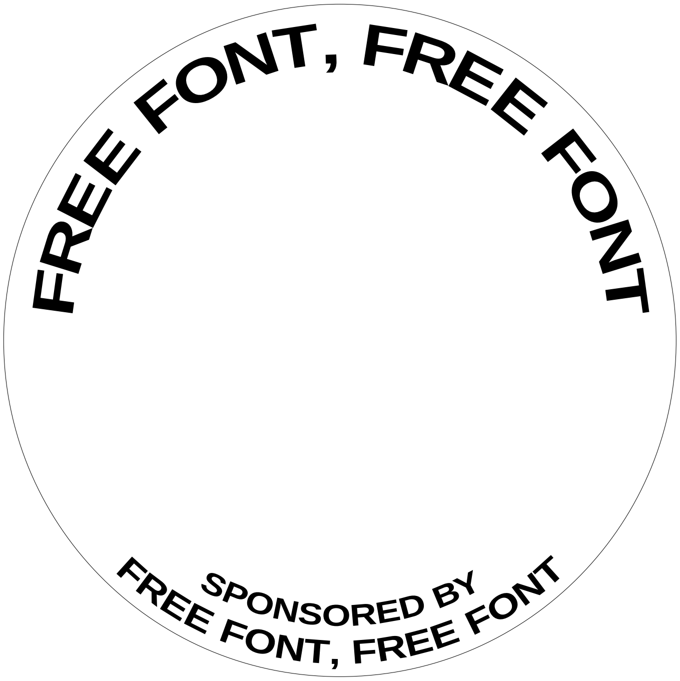
Bonus Supplement #20

Kerned arc justified font code

Kerned arc justification...

This routine let's you kern individual characters inside an arc justified message. To use, -xpos -ypos- -radius- (message with individual character justification) karcjustify. A positive radius creates upward curving arcs. A negative radius creates downward curving arcs. Use -arckern- to stretch or compress message.

```
/str ( ) def               % define a string abreviation
/customkernchar (~) def      % define a character to use for kerning

/karcjustify {gsave /msg exch def /radius exch def translate msg stringwidth pop
0 msg {customkernchar 0 get eq {1 add} if} forall dup 0 gt { customkernchar
stringwidth pop neg customkern add mul} if add msg length 1 sub arckern mul
add 2 div dup 57.29578 mul radius div msg {str exch 0 exch put gsave rotate 0
radius moveto str dup dup dup customkernchar eq not  {stringwidth pop  2 div
57.29578 mul radius div neg rotate show stringwidth pop} {customkern 2 div
57.29578 mul radius div neg rotate pop pop pop customkern} ifelse arckern add
sub dup 57.29578 mul radius div grestore} forall pop pop grestore} def

% demo, remove before use --

/arckern -1.25 def             % slight character kerning
/customkern -1.25 def
/Helvetica-Bold findfont [45 0 0 35 0 0] makefont setfont 300 305 160 (F~R~EE
F~O~N~T, F~R~EE F~O~N~T) karcjustify  % and show it convex

/customkern -1 def /arckern 0.1 def
/Helvetica-Bold findfont [22 0 0 18 0 0] makefont setfont 300 300 -170
(SPONSORED B~Y) karcjustify    % same - only concave

/customkern -0.2 def /arckern 0.05 def
/Helvetica-Bold findfont [25 0 0 20 0 0] makefont setfont 300 300 -192.5 (FR~EE
FONT, FR~EE FONT) karcjustify    % second concave line
```



For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

21

AppleTalk cables
Underlining rules
Fomeboard sheets
Pixel boundry locking
LaserWriter Reference
Popular *Star Wars* font

If your LaserWriter and host computer are plugged into the same grounded outlet, then you definitely do **not** need any special $160 AppleTalk cables. A plain old $5 small "printer" or "null modem" cable will work just fine. Note that AppleTalk is often *slower* than is stock 9600 Baud serial communications.

An exact pixel boundary lock can be done on your LaserWriter by using **300 mul 72 div floor 72 mul 300 div** for each axis. This is one way to improve a fine gray grid or to image repeating line patterns to exactly the same width. After a lock, move only in exact pixel multiples, or the lock will be lost. Exact locking is best done at 1:1 scale.

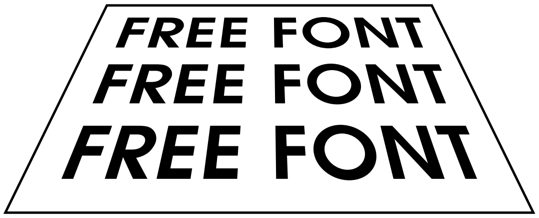
While underlining can be tolerated in the second grade, it should be strongly discouraged in the third, and severely punished in the fourth. Beyond that, anyone who even thinks about underlining in public should get staked to an anthill and left there until the next steering committee meeting. Fortunately, this disease is self limiting, since compulsive underliners eventually grow so much hair on the palms of their hands that they can no longer type.

MATERIAL OF THE MONTH —

It used to be that cardboard was cardboard and posterboard was posterboard. But today, there are dozens of **EXOTIC NEW HIGH-TECH DISPLAY MATERIALS**, also useful for signs and modelmaking. One leading source of these products is *Fomeboards*, 2211 North Elston Avenue, Chicago, IL 60614. (312) 278-9200.

PUBLICATION OF THE MONTH —

The **LASERWRITER REFERENCE MANUAL** is a new hard bound Apple book which covers the LaserWriter, LaserWriter Plus, NT and the NTX machines. Included are complete lists of those Diablo and Hewlett-Packard emulation commands, interface details, hard disk access, new PostScript commands and more. One stocking source is *Synergetics*, Box 809, Thatcher AZ, 85552. (602) 428-4073.



Everyone wants "star wars", so here is the genuine item . . .

```
/pixellineremap {0 1 pixelprocheight 300 mul 72 div cvi {/scanlinenumber
exch def save /snap1 exch def mappingproc newpath 0 scanlinenumber 72
mul 300 div moveto pixelprocwidth 0  rlineto 0 72 300 div rlineto
pixelprocwidth neg 0 rlineto closepath clip newpath pixelproc clear snap1
restore} for } def

/mappingproc {pixelprocwidth 2 div 0 translate tiltfactor pixelprocheight
mul dup scanlinenumber add div dup scale pixelprocwidth 2 div neg 0
translate} def

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

/AvantGarde-Demi findfont [40 0 0 40 0 0] makefont setfont

/pixelproc  { 5 5 moveto 0 134 rlineto 222 0 rlineto 0 -134 rlineto closepath
stroke 20 15  moveto (FREE FONT) show 20 57  moveto (FREE FONT) show
20 99  moveto (FREE FONT) show} def

/pixelprocheight 140 def  /pixelprocwidth 230 def
/tiltfactor 8 def  % the smaller the flatter

100 200 translate  pixellineremap  showpage
```

For more help:

(602) 428-4073

Don Lancaster's PostScript Secrets

22

New banner font
Display PostScript
Colorease process
Roundpath routine
Internaldict FlxProc

The **FlxProc** routine that's buried in **internaldict** is the key to showing very small typography. It accepts a pair of cubic splines on the stack and then makes them weaker for intermediate sizes and degenerates them into lines for even smaller sizings. Here's how to read it . . .

```
1183615869 internaldict begin 1183615869 internaldict /FlxProc {get}
superexec {== flush} {forall} superexec quit
```

Here's a simple routine to round the corners on any arbitrary path . . .

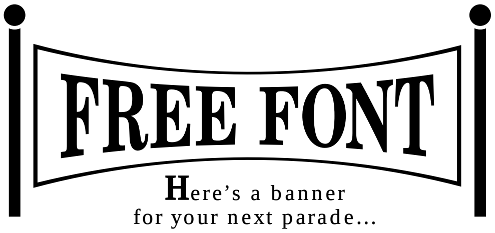
```
/roundpath { /rpdata exch def /rprad exch def rpdata
length 1 sub cvi /rppoints exch def rpdata 0 get rpdata 1
get moveto 2 2  rppoints 2 sub {/rpvalue exch def 0 1 3
{rpdata exch rpvalue add get } for rprad arcto pop pop pop
pop} for rpdata rppoints 1 sub get rpdata rppoints get
lineto} def

%  demo - remove before use

100 300 translate 10 dup scale 1 setlinecap 0.8 [12 9 8 1
0 1 10 0 5 1 0 19 0 20 5 19 10 12 10 8 9] roundpath gsave
0.6 setlinewidth 0 setgray stroke grestore gsave 1 setgray
0.4 setlinewidth stroke grestore  showpage
```

MATERIAL OF THE MONTH —

The **COLOREASE** process lets you convert laser images into multiple color real ink instant transfers that can be applied to most any surface. *Colorease*, 100 East Ohio Street, Chicago, IL, 60611. (312) 440-1266.

PUBLICATIONS OF THE MONTH —

The free **DISPLAY POSTSCRIPT OVERVIEW** and the new $30 **DISPLAY POSTSCRIPT REFERENCE MANUAL** now issued by the Adobe Systems folks, at 1585 Charleston Road, Mountain View, CA 94039. (415) 961-4400.



Here's a banner
for your next parade...

```
/vlinemap {save /plrsnap 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 slinenum == flush /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

/mapproc {50 200 translate cosrange neg degreeinc slinenum mul add cos
dup /adj exch def pixelshigh 72 mul 300 div mul dipdepth mul 2 div 0 exch
translate adj cosrange cos sub dipdepth mul 1 exch sub 1 exch scale} def

/imageproc {borderfat 2 div dup moveto 0 ptshigh borderfat sub rlineto
ptswide borderfat sub 0 rlineto 0 ptshigh neg borderfat add rlineto
closepath borderfat setlinewidth stroke 10 8 moveto stretch 0 msg ashow} def

/bannerfont {cosrange 0.48 mul ptswide div /degreeinc exch def mark 10 10
setcacheparams vlinemap} def

%  demo - remove before use

/NewCenturySchlbk-Bold findfont [30 0 0 44 0 0] makefont setfont /stretch
1 def /msg (FREE FONT) def /cosrange 60 def /dipdepth 0.8 def /ptswide 224
def /ptshigh 49 def /borderfat 3 def ptswide ptshigh vlinemap  showpage
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

NTX paper jams

Pen skip effects

Adobe Illustrator

A random screen

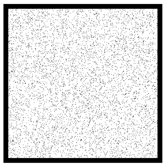
Matrocolor proofing

Printing error trapper

Those frustrating second side paper jams on your NT or NTX are caused by the pinch rollers on the upper exit tray curling the paper. For hassle free two-sided printing, either let the pages sit overnight or else use the straight through paper path and reverse your page order with software.

Here's an unusual random screen for special effects . . .

```
/random {rand 32768 div 65536 div mul floor} def

/randscreen {gsave patnum srand 300 mul 72 div /rndhigh
exch def 300 mul 72 div /rndwide exch def 72 300 div dup
scale 1 setlinecap 1 setlinewidth rndhigh rndwide mul
density mul cvi{rndwide random rndhigh random moveto
0.01 0 rlineto stroke} repeat grestore} def

/patnum 12345678 def /density 0.09 def 72 72
randscreen showpage quit
```



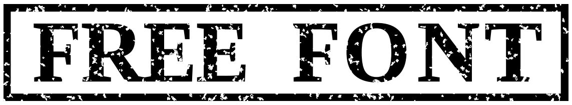
As we have already seen, a two-way environment that reports PostScript errors to your host screen is absolutely essential. Above and beyond that, you might want to go to a printing error trapper that prints your page up to the time the error occurs. Suitable error trappers appear in the Green Book, on the various *Adobe* support disks, or in my *PostScript Beginner Stuff* package. Note that the printing error trappers cannot report paper jams or similar errors.

MATERIAL OF THE MONTH —

MATROCOLOR is yet another color proofing process that lets you convert laser images into multiple color real ink instant transfers that can be applied to most any surface. From the *Castcraft Industries* folks, at 23 West Hubbard Street, Chicago, IL 60610. (312) 722-6530.

PUBLICATION OF THE MONTH —

The free **ADOBE ILLUSTRATOR** demo disk for the IBM and compatible clones, available from *Adobe Systems Inc.*, 1585 Charleston Road, Mountain View, CA 94039. (415) 961-4400.



Getting realistic looking pen skips for a curve-traced signature or for effective "splatter" screens can be very tricky. Here's a somewhat slow routine that has enough variables to do the job properly . . .

```
/random {rand 32768 div 65536 div mul floor cvi} def

/dirdict 30 dict def dirdict begin /0 {blobdist 0} def /1 {blobdist dup} def /2
{0 blobdist} def /3 {blobdist dup neg exch} def /4 {blobdist neg 0} def /5
{blobdist neg dup} def /6 {0 blobdist neg} def /7 {blobdist dup neg} def /8
{gocenter} def /9 {gocenter} def /10 {gocenter} def /11 {gocenter} def /12
{gocenter} def /13 {gocenter} def /14 {gocenter} def /15 {gocenter} def /16
{gocenter} def end /gocenter {currentpoint dup retfract mul sub neg exch
dup retfract mul sub neg exch} def

/splat {gsave splatgray setgray 72 300 div dup scale 300 mul 72 div
/splathigh exch def blobscale div 300 mul 72 div /splatwide exch def
patnum srand 1 setlinecap 1 setlinejoin 1 setlinewidth numsplats cvi {gsave
newpath splatwide random splathigh random translate 0 0 moveto
maxsplatlength random {numrandoptions 1 add cvi random (   ) cvs cvn
dirdict exch get exec rlineto } repeat stroke grestore} repeat grestore} def

200 300 translate /Palatino-Bold findfont [36 0 0 36 0 0] makefont setfont
12 8 moveto (FREE  FONT) show /numsplats 1000 def /patnum 44444444
def /retfract 0.3 def /numrandoptions 9 def /maxsplatlength 60 def
/blobscale 1 def /blobdist 1 def /splatgray 1 def 235 40 splat showpage
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #23-A

Scribbling
along a path

Scribbling in PostScript...

Sometimes the problem with PostScript is that it is TOO precise. There are times and places where you want to pick up a "hand drawn" or "sketched" effect, where a line only "sort of" followes a line or a curved path. Or where you want "slightly ratty" results.

The following PRELIMIINARY tools should get you started. Note that these tools are far more powerful in a level 2 environment, but they certainly can be used in level 1. Watch your total path length in level 1 per the IMPORTANT NOTE below.

```
%  A SCRIBBLE UTILITY TOOLKIT

%  /random is borrowed from the gonzo powertools. It returns a random integer
%  as in 6 random gives you a value from 0 to 5...

/random {rand 65536 div 32768 div mul cvi} def

%  /scribblepat sets the exact pattern used, assuming you want it to be the same
%  for each and every copy. Changing scribblepat changes when and where the
%  wiggles appear. Use any integer...

/scribblepat 12345 def     %  set fuzzy pattern

scribblepat srand          %  place this line at the start of your FIRST
                        %  USE ONLY of the scribbling routines.

%  /scribbleres determines how long each straight line approximation to the fuzzy
%  line will be. Use 1 for one point increments or 4.16667 or 0.24 for single
%  pixels. The smaller the number and the longer the line, the longer your path.
%  This can cause problems on level 1...

/scribbleres 1 def         %  length of shortest scribble

%  /ratticity determines how violently the fuzzy line deviates from the intended
%  path. It is expressed as a fraction of scribbleres. Use 0.2 for a fuzzy line, 5
%  for a stock graph. Experiment...

/ratticity 0.2 def         %  how erratic is the line to be?

%  /hominginstinct determines how motivated the ratty line will be towards
%  getting back on path. This is really a "high pass filter" or a "house take" that
%  prevents major deviations from piling up on a long term basis.

%  You can think of this as an "excess deviation tax" Normal range is 0 to 0.3
%  (zero to thirty percent) with a value of 0.3 being "get back on track fast"
%  and 1.0 being "wander all you want".

/hominginstinct 0.01 def    % how fast does the line get back on track?

%  /scribblerlineto works just like rlineto, except that an erratic path is
%  generated...

/scribblerlineto {currentpoint 2 copy translate /cury exch store /curx exch def 2
copy exch atan dup /curang exch store rotate dup mul exch dup mul add sqrt
/fulllength exch store fulllength scribbleres div ceiling cvi /numsegs exch def
fulllength numsegs div /incr1 exch store /deltay 0 store numsegs {incr1 101
random 50 sub dup 0 eq {pop 0.00001}if 50 div scribbleres ratticity mul mul
currentpoint exch pop hominginstinct mul neg add rlineto} repeat curang neg
rotate curx neg cury neg translate} def
```

IMPORTANT NOTE: Level 1 machines only allow 1500 or so elements in any path and may overflow on fine detail or long scribbles. A workaround is to place an INTERNAL *** currentpoint stroke moveto *** INSIDE OF the above routine. This will allow ultra fancy or ultra long scribbling on a level 1 machine, but will prohibit fancy path fills, clips, or multiple strokes. It will also execute MUCH more slowly. With internal stroking, the demos will have to be modified so that they do not stroke or fill by themselves.

For more help:
(602) 428-4073

Scribbling in PostScript...

```
%  /scribblecurveres uses setflat and flattenpath to approximate the curve as
%   line segments. Allowable values in pixels are 0.2 to 100. The larger the value,
%   the greater the deviation from the curve, but the faster the execution time...

/scribblecurveres 5 def        % allowable curveto path deviation

%  /scribblecurveto works just like curveto, except that an erratic path is
generated that more or less follows the curve...

/scribblecurveto {gsave currentpoint newpath moveto scribblecurveres setflat
curveto flattenpath mark { }{ }{ }{ } pathforall] /cpath exch store grestore 0 1 cpath
length 2 sub 2 div 1 sub cvi {2 mul /posn exch def cpath posn get /cx0 exch store
cpath posn 1 add get /cy0 exch store cpath posn 2 add get /cx1 exch store cpath
posn 3 add get /cy1 exch store cx1 cx0 sub cy1 cy0 sub scribblerlineto} for} def

%  //// demo - remove or alter before reuse. ///

0 100 translate   % improve position on page
```

(A) A PLAIN OLD "HAND DRAWN" LINE

```
% This shows you how to draw a fuzzy line....

/hominginstinct 0.3 def   % how fast does the line get back on track?
/scribbleres 1 def        % length of shortest scribble
/ratticity  1 def         % how erratic is the line to be?

1 setlinewidth            % set width of stroked line

100 200 moveto            % set initial position of line
200 0 scribblerlineto     % and draw the fuzzy line

stroke                    % complete the path
```

(B) A FAKE STOCK MARKET PRICE HISTORY

```
% This shows you how to draw an erratic graph ....

/hominginstinct 0 def     % how fast does the line get back on track?
/scribbleres 1 def        % length of shortest scribble
/ratticity  6 def         % how erratic is the line to be?

0 setlinewidth            % set width of stroked line

100 300 moveto            % set initial position of line
200  0 scribblerlineto    % and draw the fuzzy line

stroke                    % and complete the path
```

(C) A SLIGHTLY RATTY BORDER

```
% This shows you how to draw a slightly ratty border ....

/hominginstinct 0.3 def   % how fast does the line get back on track?
/scribbleres 1 def        % length of shortest scribble
/ratticity  0.8 def       % how erratic is the line to be?

1 setlinecap              % smooth line closures
1 setlinejoin             % ditto
6 setlinewidth            % set width of stroked line

150 350 moveto            % set initial position of line
100  0 scribblerlineto    % and draw the box bottom
0 133 scribblerlineto     %          right side
-100 0 scribblerlineto    %          top
0 -133 scribblerlineto    %          left side

stroke                    % and draw the line
```

Don Lancaster's PostScript Secrets

Bonus Supplement #23-B

Scribbling
along a path

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

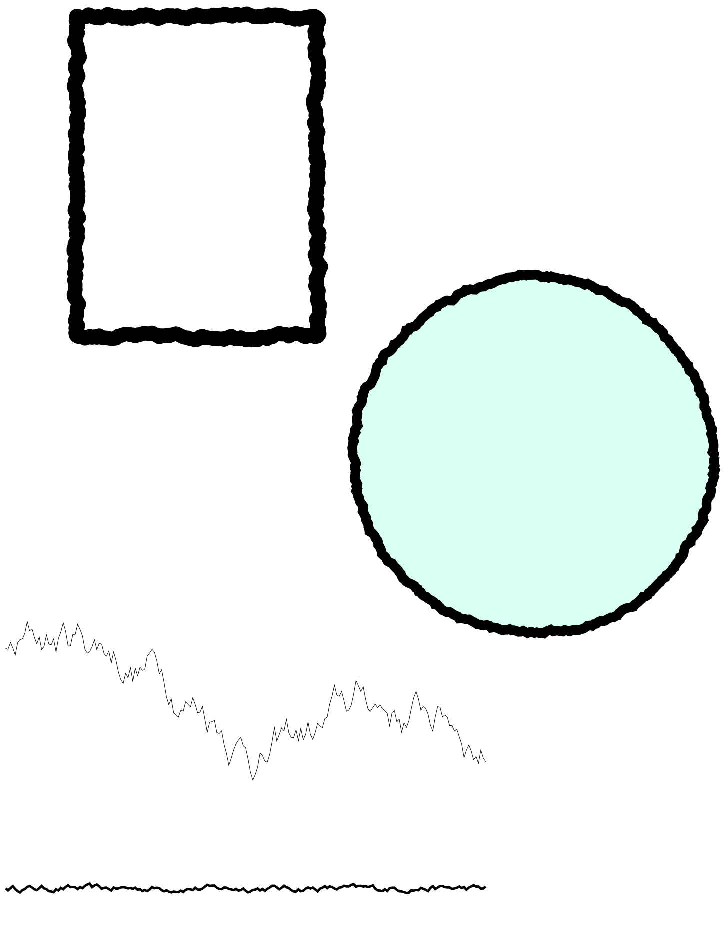
Bonus Supplement #23-c

Scribbling along a path

Scribbling in PostScript...

(D) DRAWING A FILLED FUZZY CIRCLE

```
-30 10 translate          % improve position on page

106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen % nice gray

/scribblecurveres 5 def    % allowable curveto path deviation
/hominginstinct 0.4 def    % how fast does the line get back on track?
/scribbleres 1 def         % length of shortest scribble
/ratticity   0.5 def       % how erratic is the line to be?

4 setlinewidth             % set width of stroked line

525 350 moveto             % initial position

525 391.4 491.4 425 450 425  scribblecurveto   % first quadrant
408.6 425 375 391.4 375 350  scribblecurveto   % second quadrant
375 308.6 408.6 275 450 275  scribblecurveto   % third quadrant
491.4 275 525 308.6 525 350  scribblecurveto   % fourth quadrant

gsave 0.95 setgray fill grestore stroke        % outline and fill

showpage
```



For more help:
(602) 428-4073

Do NOT ever use a parallel output port on your IBM or clone to drive your LaserWriter or other PostScript laser printer! Always use the serial COM ports. Besides blatantly violating PostScript's device independence, most parallel drivers prevent you from properly receiving error messages and prohibit you from recording compiled output or any other results sent back from your LaserWriter to the host computer. In addition, the higher serial baud rates are often faster than parallel comm standards.

Apple definitely does not want you to know about the incredibly powerful and zero cost two-host network that always has been built into each and every LaserWriter. This network does not require $150 cables, does not use AppleTalk, can be ridiculously faster than AppleTalk, and needs no A-B switch box or manual intervention. Simply use the serial comm mode and connect your first host to the 25 pin connector and the second one to the 8 or 9 pin connector. The LaserWriter will automatically take turns accepting any job from any input, and each host can make a status query at any time. You can use the same idea for dual speed printing from one host, using 19200 baud for interactive and 57600 baud for any book-on-demand printing of compiled code.

Second side NT or NTX paper feed jams can often be traced to slick or worn pickup rollers or registration pads. Do NOT ever use alcohol to clean these; use a good office machine platen cleaner and reconditioner. These can need replacement every 25,000 or so copies. On the Apple LaserWriter, you can use Hewlett-Packard parts #RG1-0931-00CN (roller), and #RF1-1145-00CN (pad), following the instructions in H-P's manual #33440-90904. The free H-P phone number is (800) 227-8164.

MATERIALS OF THE MONTH —

TICKET BLANKS, both plain and numbered are available from *Paper Plus* (213) 436-8291, *Blanks USA* (800) 328-7311, *Ticket Express* (800) 544-8520, and *Quick Tickets* (800) 521-1142. *Blanks USA* has a free sample pack of 300 tickets available. Meanwhile, the world's cheapest ticket perforator is available from *JerryCo* (312) 475-8440 as stock 3620 for $2.00. It is cleverly disguised as a pair of Airedale pruning shears. A sewing machine also works.

PUBLICATION OF THE MONTH —

The pricey but very informative **BOVE & RHODES INSIDE REPORT** gives lots of outstanding PostScript and laser printing info, especially for all those high end applications. $195 per year from *Bove & Rhodes*, Box 1289, Gualala CA, 95445. (707) 884-4413.

FREE FONT

Suitable tampering with the font matrix can give you an astounding range of special effects. In this very simple example, we have stretched out plain old *Helvetica-Bold* to where it almost will assume *Copperplate* or even *Steelplate* vibes . . .

```
/Helvetica-Bold findfont [60 0 0 30 0 0] makefont setfont
200 300 moveto (FREE  FONT) show showpage
```

More generally, if you use an **[aa bb cc dd ee ff]** font matrix, then . . .

Value **aa** – is how fat and sets character **width**
Value **bb** – is how much special effect **climb** (usually zero)
Value **cc** – is how much **lean** for italics (usually zero)

Value **dd** – is how tall and sets character **height**
Value **ee** – is how much **horizontal shift** (usually zero)
Value **ff** – is how much **vertical shift** (for superscripts or subscripts)

Don Lancaster's PostScript Secrets

24

Ticket Blanks
A Copperplate font
Serial comm network
IBM serial COM ports
NTX second side jams
Bove & Rhodes Report

For more help:
(520) 428-4073

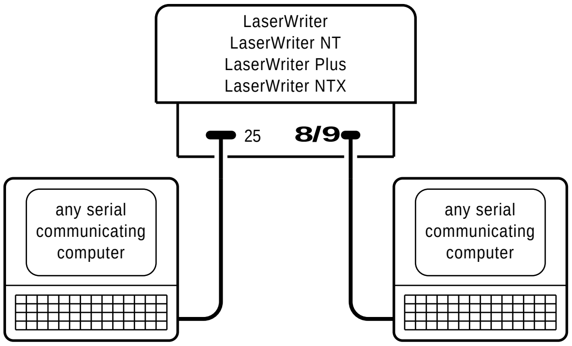
Don Lancaster's PostScript Secrets

Bonus Supplement #24

Three Zero Cost LaserWriter Networks

For more help:
(520) 428-4073

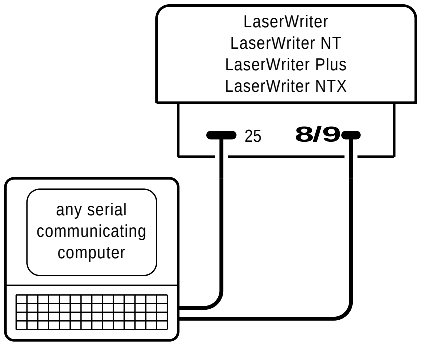
Three secret LaserWriter networking schemes . . .


LaserWriter
LaserWriter NT
LaserWriter Plus
LaserWriter NTX
25
8/9
any serial communicating computer
any serial communicating computer

In a **Type I** network, two (or more) serially communicating computers are used. Each automatically takes turns using the network and each runs at its own optimum baud rate.

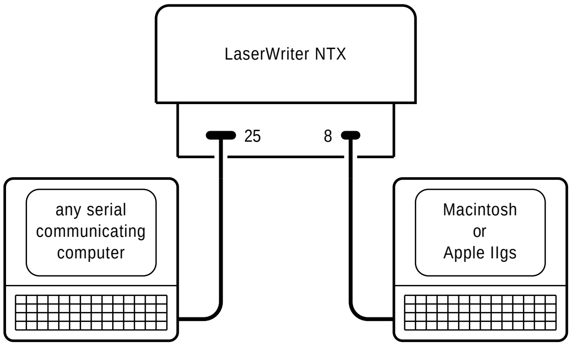
Either computer can interrogate the LaserWriter status at any time by use of the usual **[control]-t** command.

A simple A-B switching box can be added to either side to extend the network to any number of computers.



In a **Type II** network, one serially communicating computer is routed to the LaserWriter by way of two serial data channels. The first channel is a normal 9600 or 19200 baud full duplex comm line. The second channel is a hand crafted, one-way line running at an honest 57600 baud.

The second channel is used for rapid downloads of known good code, font and prolog downloads, and for use when book-on-demand publishing. A printing error trapper is needed when using this channel.



In a **Type III** network, one serial communicating computer gets used, along with a Mac or IIgs switchable between serial and AppleTalk. The Mac or IIgs **must** use an ordinary printer cable and **must** be plugged into the same grounded ac outlet as the LaserWriter.

Creative use of the **0 sethardwareiomode** (software switch to serial) and **2 sethardwareiomode** (software switch back to AppleTalk) is used to avoid any physical switch flipping. This scheme only works on the LaserWriter NTX.

Don Lancaster's PostScript Secrets

25

To switch your LaserWriter NTX from serial to AppleTalk, you can use the **2 sethardwareiomode** command sent serially. At the end of your present job, you will be in AppleTalk, and the serial channel will end up trashed and useless. Either serial channel can send this command.

Similarly, to switch your LaserWriter NTX from AppleTalk back to Serial, use the **0 sethardwareiomode** command sent over AppleTalk. At the end of the present job, both serial channels will end up active, and your AppleTalk will get disconnected. Any custom serial baud rates will be preserved so long as the actual DIP switches are not flipped.

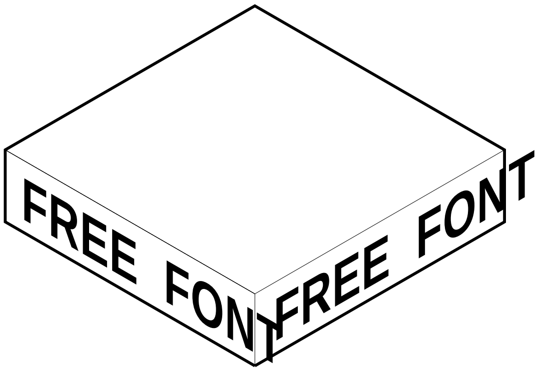
There is an extremely bad bug in AppleTalk that blatantly violates PostScript's device independence. Files sent over AppleTalk are treated differently than files sent serially, with possible disastrous results. A serial file correctly changes all carriage returns to newline characters; AppleTalk does not. Sledgehammer cures involve never using *readline*, or else having your custom input scanner code trap out both the carriage return and newline characters.

MATERIAL OF THE MONTH —

While the best-known source of **BADGE PARTS** is *Badge-A-Minit*, 348 North 30th Road, Box 800, LaSalle IL 61301 (815) 224-2090, you'll find much better prices at *Super Badge and Button Supplies*, at 2338 West Burnham Street, Milwaukee, WI 53204. (800) 328-8272.

PUBLICATION OF THE MONTH —

The "orange book", and otherwise known as **REAL WORLD POSTSCRIPT**, is newly published by *Addison Wesley*. While not an Adobe book, and not even remotely in the same league as the red and blue books, this does have interesting and useful material on color separations, halftoning, and font encoding. One source is *Synergetics* (602) 428-4073.



Isometric lettering is handy for engineering drawings or "open" book or menu pages, and is extremely fast and simple to do . . .

```
/pointsize 40 def  /Helvetica-Narrow-Bold findfont [pointsize dup .577 mul 0
pointsize 0 0] makefont setfont 100 300 moveto 1 0 (FREE FONT) ashow
%right side lettering

/pointsize 40 def  /Helvetica-Narrow-Bold findfont [pointsize dup -.577 mul 0
pointsize 0 0] makefont setfont 100 300 moveto 1 0 (FREE FONT) ashow
%left side lettering
showpage quit
```

The remaining ten isometric font positions are handled similarly. A complete set of isometric drawing routines appears in my *PostScript Show and Tell*.

For more help:

(602) 428-4073

Don Lancaster's PostScript Secrets

26

Compiled PostScript

The *Printer's Shopper*

Non-woven interfacing

Superinsidestroke font

PostScript superstroking

All of those long page makeready times on your LaserWriter can be eliminated outright or else greatly minimized by going to a PostScript compiling process. To do this, you redefine *awidthshow* and the other operators so that they return only the essential final results back to your host for recording and later reuse. See my *Ask The Guru II* for full details.

Two elegant tools, **superstroke** and **superinsidestroke** can improve any outlined borders, pipes, wires, or create unusual font effects...

```
/superstroke { save /sssnap exch def /sscmd exch def 0 2 sscmd
length 2 div cvi 1 sub 2 mul {/aposn exch def gsave sscmd aposn
get setlinewidth sscmd aposn 1 add get setgray stroke grestore}
for clear sssnap restore newpath} def

/superinsidestroke {save clip /sssnap exch def /sscmd exch def
mark 0 2 sscmd length 2 div cvi 1 sub 2 mul {/aposn exch def
gsave sscmd aposn get 2 mul setlinewidth sscmd aposn 1 add
get setgray stroke grestore} for cleartomark sssnap restore
newpath} def

% demo - remove before use...

100 200 moveto 0 100 rlineto 300 0 rlineto 0 -100 rlineto
closepath [9 0 7.5 0.8 3 0] superinsidestroke  showpage quit
```

Each pair of the values in your array decides a linewidth and a grey level. You always start with the *widest* value pairs. A **superstroke** works symetrically, while **superinsidestroke** works only *inside* the original path.

MATERIAL OF THE MONTH —

PELLON, otherwise known as **NON-WOVEN INTERFACING**, may be safely run on through your LaserWriter NT or NTX. It costs a buck a yard at your local cloth or notions store and can become a highly useful transfer, stencil, and pattern material. WARNING: Be absolutely sure to get the "sew-in" style and NOT the "heat fusable" type! The medium thickness works best.

PUBLICATION OF THE MONTH —

THE PRINTER'S SHOPPER is not really a shopper. It is instead a company catalog that is chock full of a mind boggling variety of printshop materials and tools useful to all your small scale laser printing and desktop publishing. Box 1056, Chula Vista, CA 92012 (619) 422-2200.

FREE FONT

Superstroking can also give you all sorts of special font effects...

```
% requires superinsidestroke from above

/strokeproc {[5 0 1 1 0.5 0] superinsidestroke} def /sstr (X) def /extrastretch
2 def

/superfont {gsave /msg exch def msg {/cval exch def save /csnap exch def
currentpoint sstr 0 cval put sstr false charpath strokeproc csnap restore
exch sstr stringwidth pop add extrastretch add exch moveto} forall grestore}
def

% demo- remove before use...

1 setlinecap 1 setlinejoin 100 200 moveto
/Bookman-DemiItalic findfont [40 0 0 40 0 0] makefont setfont

(FREE FONT) superfont  showpage
```

Use **superinsidestroke** whenever you want to preserve the font outline, and the **superstroke** when you want to fatten the original outline.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #26-A

Sleezoid utility

Avuncular sleezoid utility...

There are times and places when you want to relate the graph space of a Bezier cubic spline to its equation space and vice versa. Among other things, this lets you move anywhere along any portion of a SINGLE curveto cubic spline. Which can lead to such exotics as controlled distortion.

In graph space, a Bezier curve starts at x0,y0 and ends at x3 y3. It leaves x0,y0 with a direction and an enthuasiasm set by the first control point x1,y1 and enters x3,y3 with a direction and an enthuasiasm set by the second control point x2,y2.

Here are the same Bezier formulas in the algebraic equation space:

```
x(t) = A(t^3) + B(t^2) + C(t) + x0
y(t) = D(t^3) + E(t^2) + F(t) + y0
```

Note that (t) varies from 0 to 1 from the beginning to the end of your spline curve.

Here is how to get from equation space to graph space:

```
x1 = x0 + C/3            y1 = y0 + F/3
x2 = x1 + C/3 + B/3      y2 = y1 + F/3 + E/3
x3 = x0 + C + B + A      y3 = y0 + F + E + D
```

And here is how to get from graph space back to equation space ..

```
A = x3 - 3x2 + 3x1 + x0    D = y3 - 3y2 + 3 y1 + y0
B = 3x2 - 6x1 + 3x0        E = 3y2 - 6y1 + 3y0
C = 3x1 - 3x0              F = 3y1 - 3y0
```

Our demo shows an avuncular sleezoid, rather than the older transverse lenticular variation. See H.K. Hornswoggle and N.V. Blatenworth's classic EXPLOITING SLEEZOIDS FOR FUN AND PROFIT, Brandenburg Press, Boise, ID, 1964, pages 55-27500 for a thorough, but definitely unterse, treatment.

Actually, the first half of this is in the red book, and the vastly more useful second half follows by simple ninth grade algebra.

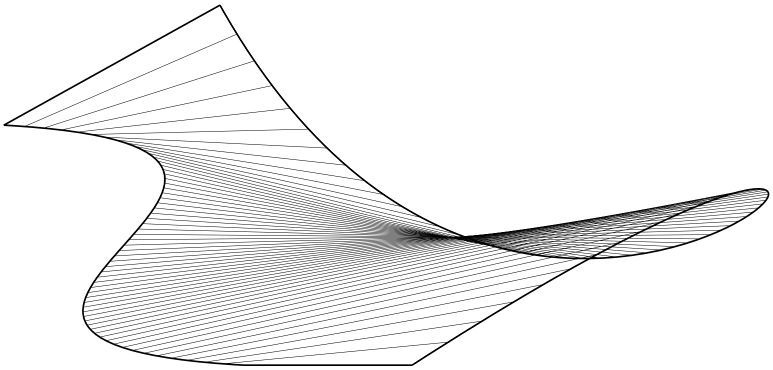
Here's how to do it in PostScript ...

```
% pcurveto1 grabs the parameters for a curveto

/pcurveto1 {6 copy /y3 exch def /x3 exch def /y2 exch def /x2 exch def /y1 exch
def /x1 exch def currentpoint /y0 exch def /x0 exch def curveto} def

% this finds x(t) given t in the range of 0 to 1 ...

/xtt {x3 x2 3 mul sub x1 3 mul add x0 sub tt 3 exp mul x2 3 mul x1 6 mul neg
add x0 3 mul add tt dup mul mul add x1 3 mul x0 3 mul neg add tt mul add x0
add} def

% this finds y(t) given t in the range of 0 to 1 ...

/ytt {y3 y2 3 mul sub y1 3 mul add y0 sub tt 3 exp mul y2 3 mul y1 6 mul neg add
y0 3 mul add tt dup mul mul add y1 3 mul y0 3 mul neg add tt mul add y0 add}
def

% pcurveto2 does the same thing for a second and separate curve

/pcurveto2 {6 copy /yy3 exch def /xx3 exch def /yy2 exch def /xx2 exch def /yy1
exch def /xx1 exch def currentpoint /yy0 exch def /xx0 exch def curveto} def

/xxtt {xx3 xx2 3 mul sub xx1 3 mul add xx0 sub tt 3 exp mul xx2 3 mul xx1 6 mul
neg add xx0 3 mul add tt dup mul mul add xx1 3 mul xx0 3 mul neg add tt mul
add xx0 add} def

/yytt {yy3 yy2 3 mul sub yy1 3 mul add yy0 sub tt 3 exp mul yy2 3 mul yy1 6 mul
neg add yy0 3 mul add tt dup mul mul add yy1 3 mul yy0 3 mul neg add tt mul
add yy0 add} def
```

For more help:
(520) 428-4073

Avuncular sleezoid utility...

```
% if you want more curves, use pcurveto3, etc...

/numpoints 25 def % default

/dot { currentpoint newpath 0.150 0 360 arc fill } def % from GUTILITY.PTL

/plotdots {0 1 numpoints div 1 {/tt exch def xtt ytt moveto dot} for }def

/plotsurface {0 1 numpoints div 1.0001 {/tt exch def xtt ytt moveto xxtt yytt
lineto stroke} for } def

%  //// demo - remove before use ////

gsave                 % save graphics state
45 450 translate      % position on page
16 dup scale          % set size
-90 rotate            % orientatify

0.07 setlinewidth     % set thickness

0 0 moveto 1 18 9 -8 10 10 pcurveto1 stroke      % draw first spline
-5 9 moveto 20 23 -10 48 10 17 pcurveto2 stroke  % draw second spline

0 0 moveto -5 9 rlineto stroke                   % far edge trim
10 10 moveto 10 17 lineto stroke                 % near edge trim

/numpoints 60 def                                % set spacing
0 setlinewidth plotsurface                       % connect the dots

grestore                                         % restore state

showpage quit                                    % and end it all
```



Don Lancaster's
PostScript
Secrets

Bonus Supplement #26-B

Sleezoid
utility

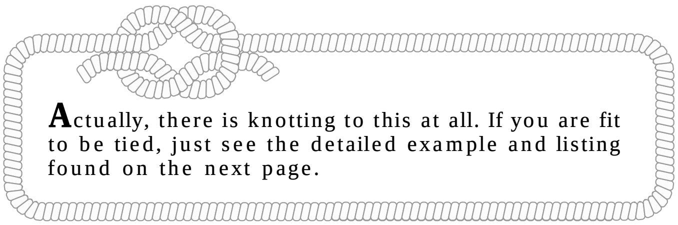
For more help:
(520) 428-4073

That non-PostScript LaserWriter SC seems to be a lousy buy. The $1500 street price H-P LaserJet II does far more with the same Canon SX engine. Worse still, whenever your *inevitable* SC upgrade to the PostScript NT or NTX is made, you will be fined an "upgrading penalty" that can exceed $3000. If you can't affort a LaserWriter NT, buy a NEC LC890 instead. Good pricing on the NT and NTX is found at (818) 376-1662, while LC890 pricing is found at (800) 221-7774.

Don Lancaster's PostScript Secrets

27

Outline-and-Fill font
LaserWriter SC ripoff
Quick Printing journal
Sneaky bitmap access
Knotting and macrame
Drum recoating service



Actually, there is knotting to this at all. If you are fit to be tied, just see the detailed example and listing found on the next page.

Accessing PostScript bitmaps can be tricky, but often will ridiculously speed up certain jobs. Three scams that can work do include: (1) use **copypage** instead of **showpage** and erase only the stuff that changes for flyers, shipping labels, or business letters; (2) Make your logo or other slow-executing routine into a font and let the font cache convert it into a bit map for you (very handy for 12-up business cards); and (3) The font caches on a NTX can be read from the FC/ subdirectory of your hard disk and then be host reprocessed.

MATERIALS OF THE MONTH —

An **SX DRUM HARD RECOATING SERVICE** is now available. The early reports claim eight and even nine SX reloads. Contact Arlin Shepard at *Lazer Products* (303) 792-5277 for full details. A great SX and CX **PIN PULLING TOOL** is available as the #AXP43-001 *GlompenStractor* from Don Thompson at (714) 855-3838. Two fairly low cost and stocking sources of **BRAND NEW SX CARTRIDGES** are (800) 228-6637 and (603) 669-1641.

PUBLICATION OF THE MONTH —

QUICK PRINTING is a free trade journal for the instant printing folks, chock full of unusual material and supply sources. *Coast Publishing* 1680 SW Bayshore Blvd, St. Lucie, FL 34964. (407) 879-6666.

FREE FONT

The two most common mistakes made when trying to do an outline-and-fill are (1) failing to select a non-putrid gray, and (2) failing to clip to the exact outline of each character. Here's how its done . . .

```
% requires superinsidestroke from LaserWriter Secrets #26

/strokeproc {[9 0.9 0.3 0] superinsidestroke} def
/sstr (X) def /extrastretch 2 def
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen

/superfont {gsave /msg exch def msg {/cval exch def save /csnap exch def
currentpoint sstr 0 cval put sstr false charpath strokeproc csnap restore
exch sstr stringwidth pop add extrastretch add exch moveto} forall grestore}
def

% demo- remove before use . . .

1 setlinecap 1 setlinejoin 100 200 moveto
/Palatino-Bold findfont [40 0 0 40 0 0] makefont setfont

(FREE  FONT) superfont  showpage  quit
```

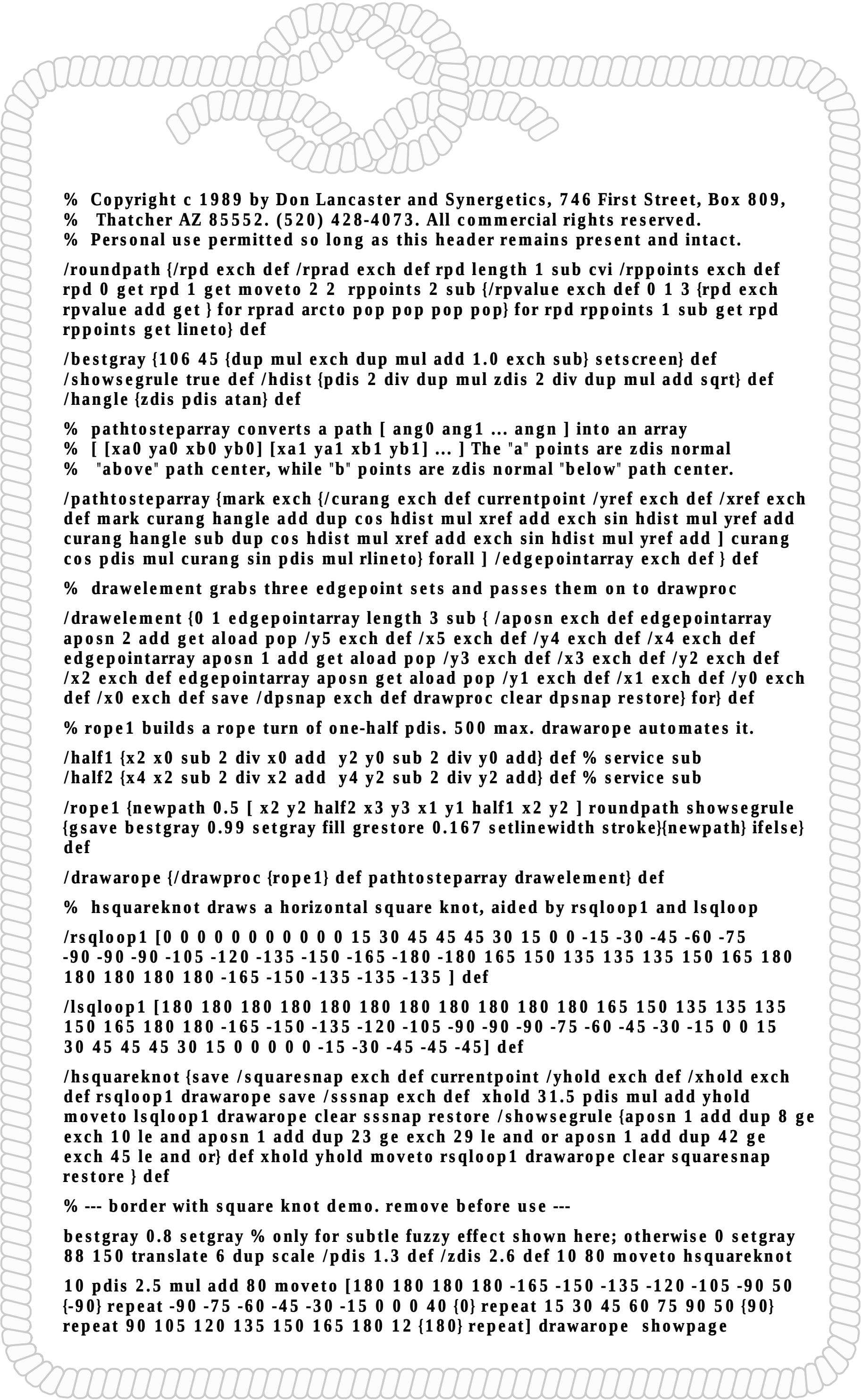
For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #27

Knots to You, Charlie

There's knotting to this at all . . .



```
%  Copyright c 1989 by Don Lancaster and Synergetics, 746 First Street, Box 809,
%   Thatcher AZ 85552. (520) 428-4073. All commercial rights reserved.
%  Personal use permitted so long as this header remains present and intact.

/roundpath {/rpd exch def /rprad exch def rpd length 1 sub cvi /rppoints exch def
rpd 0 get rpd 1 get moveto 2 2  rppoints 2 sub {/rpvalue exch def 0 1 3 {rpd exch
rpvalue add get } for rprad arcto pop pop pop pop} for rpd rppoints 1 sub get rpd
rppoints get lineto} def

/bestgray {106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen} def
/showsegrule true def /hdist {pdis 2 div dup mul zdis 2 div dup mul add sqrt} def
/hangle {zdis pdis atan} def

%  pathtosteparray converts a path [ ang0 ang1 ... angn ] into an array
%  [ [xa0 ya0 xb0 yb0] [xa1 ya1 xb1 yb1] ... ] The "a" points are zdis normal
%   "above" path center, while "b" points are zdis normal "below" path center.

/pathtosteparray {mark exch {/curang exch def currentpoint /yref exch def /xref exch
def mark curang hangle add dup cos hdist mul xref add exch sin hdist mul yref add
curang hangle sub dup cos hdist mul xref add exch sin hdist mul yref add ] curang
cos pdis mul curang sin pdis mul rlineto} forall ] /edgepointarray exch def } def

%  drawelement grabs three edgepoint sets and passes them on to drawproc

/drawelement {0 1 edgepointarray length 3 sub { /aposn exch def edgepointarray
aposn 2 add get aload pop /y5 exch def /x5 exch def /y4 exch def /x4 exch def
edgepointarray aposn 1 add get aload pop /y3 exch def /x3 exch def /y2 exch def
/x2 exch def edgepointarray aposn get aload pop /y1 exch def /x1 exch def /y0 exch
def /x0 exch def save /dpsnap exch def drawproc clear dpsnap restore} for} def

% rope1 builds a rope turn of one-half pdis. 500 max. drawarope automates it.

/half1 {x2 x0 sub 2 div x0 add  y2 y0 sub 2 div y0 add} def % service sub
/half2 {x4 x2 sub 2 div x2 add  y4 y2 sub 2 div y2 add} def % service sub

/rope1 {newpath 0.5 [ x2 y2 half2 x3 y3 x1 y1 half1 x2 y2 ] roundpath showsegrule
{gsave bestgray 0.99 setgray fill grestore 0.167 setlinewidth stroke}{newpath} ifelse}
def

/drawarope {/drawproc {rope1} def pathtosteparray drawelement} def

%  hsquareknot draws a horizontal square knot, aided by rsqloop1 and lsqloop

/rsqloop1 [0 0 0 0 0 0 0 0 0 0 0 15 30 45 45 45 30 15 0 0 -15 -30 -45 -60 -75
-90 -90 -90 -105 -120 -135 -150 -165 -180 -180 165 150 135 135 135 150 165 180
180 180 180 180 -165 -150 -135 -135 -135 ] def

/lsqloop1 [180 180 180 180 180 180 180 180 180 180 180 165 150 135 135 135
150 165 180 180 -165 -150 -135 -120 -105 -90 -90 -90 -75 -60 -45 -30 -15 0 0 15
30 45 45 45 30 15 0 0 0 0 0 -15 -30 -45 -45 -45] def

/hsquareknot {save /squaresnap exch def currentpoint /yhold exch def /xhold exch
def rsqloop1 drawarope save /sssnap exch def  xhold 31.5 pdis mul add yhold
moveto lsqloop1 drawarope clear sssnap restore /showsegrule {aposn 1 add dup 8 ge
exch 10 le and aposn 1 add dup 23 ge exch 29 le and or aposn 1 add dup 42 ge
exch 45 le and or} def xhold yhold moveto rsqloop1 drawarope clear squaresnap
restore } def

% --- border with square knot demo. remove before use ---

bestgray 0.8 setgray % only for subtle fuzzy effect shown here; otherwise 0 setgray
88 150 translate 6 dup scale /pdis 1.3 def /zdis 2.6 def 10 80 moveto hsquareknot

10 pdis 2.5 mul add 80 moveto [180 180 180 180 -165 -150 -135 -120 -105 -90 50
{-90} repeat -90 -75 -60 -45 -30 -15 0 0 0 40 {0} repeat 15 30 45 60 75 90 50 {90}
repeat 90 105 120 135 150 165 180 12 {180} repeat] drawarope  showpage
```

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

28

NTX disk blowups

Signmaking catalog

Laser to litho plates

True perspective font

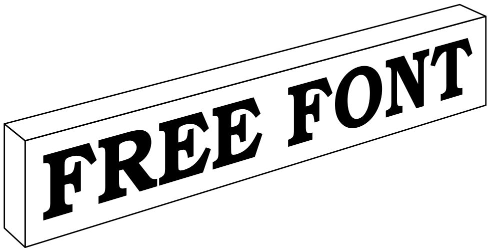
The disk operating system for the NTX is incredibly flakey and will blow up if you so much as think about looking at it sideways. *Always* have a host based NTX hard disk rebuilding program on hand and *always* stay 100 percent backed up at all times. Be certain to budget at least three hours per week for your NTX hard disk rebuilding and maintenance.

MATERIAL OF THE MONTH —

Direct **LASER TO LITHO PRINTING PLATES** can let you use any laser printer as a platemaker for traditional litho printing, at 20 cents a plate and involving zero chemistry or any other makeready. Each plate is good for a thousand copies in real ink of any color, including thermography. *REL Graphics* at (800) 521-1080, or from *Dominick Argenio* at (800) 631-0039.

PUBLICATION OF THE MONTH —

Dick Blick's **SIGN MAKING AND SCREEN PRINTING CATALOG** is crammed full of unusual materials, some laser printable and some not. These do include vinyl bumpersticker stock, fluorescents, glow-in-the-darks, the *Coburn* and *Scotchlite* films, signboards, plus bunches more. From *Dick Blick* at Box 1267, Galesburg, IL 61401. (800) 447-8192.



Here's a slow but effective method to put all your PostScript lettering into a whole new true two-point 3-D perspective . . .

```
/px {/zz exch def /yy exch def /xx exch def yo dup yy add div dup xx xo sub
mul exch zz zo sub mul} def /psave {/zh zz def /yh yy def /xh xx def} def
/prestore {/zz zh def /yy yh def /xx xh def xx yy zz prm} def /psave1 {/zh1 zz
def /yh1 yy def /xh1 xx def} def /prestore1 {/zz zh1 def /yy yh1 def /xx xh1
def  xx yy zz prm} def

/pm {px psave moveto} def /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 /prmtostack {/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} def

/pd {px psave lineto} def /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 /xrd {0 0 prd} def /xrm {0 0 prm}
def /yrd {0 exch 0 prd} def /yrm {0 exch 0 prm} def /zrd {0 exch 0 exch prd}
def /zrm {0 exch 0 exch prm} def

/pixellineremap {0 1 pxpwidth 300 mul 72 div cvi {/sline# exch def save
/snap1 exch def gsave mappingproc newpath sline# 72 mul 300 div 0
moveto 0 pxpheight rlineto 0 0 rlineto 0 pxpheight neg rlineto closepath
clip newpath pixelproc grestore clear snap1 restore} for } bind def

/mappingproc {10 sline# 72 mul 300 div add 0 16 prmtostack 2 copy /ybot
exch def pop exch sline# 72 mul 300 div sub exch dtransform ceiling exch
ceiling exch idtransform translate 10 sline# 72 mul 300 div add 0 16
pxpheight add prmtostack exch pop ybot sub pxpheight div 1 exch scale}
bind def

% demo - remove before use . . .

160 700 translate 0 setlinewidth /xo 00 def /yo 700 def /zo 400 def /objrot
30 def 0 0 0 pm 50 zrd 300 xrd -50 zrd -300 xrd 20 yrd 50 zrd -20 yrd 20
yrd 300 xrd -20 yrd 0.6 setlinewidth stroke

/Bookman-Demi findfont [45 0 0 45 0 0] makefont setfont mark 10 10
setcacheparams /pixelproc {0 2 moveto 0.4 0 (FREE FONT) ashow} def
/pxpwidth 300 def /pxpheight 35 def -1 0 -9 pm pixellineremap showpage
```

For more help:

(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #28-A

Adobe timing utilities

Adobe timing utilities...

Four of the Adobe font timing modules have been gathered together into one single file for convenience in downloading.

Timing modules let you measure how long portions of your code are taking, letting you put your speedup techniques where they will do the most good.

```
%!PS-Adobe-3.0 ExitServer
%%Title: Download Interval Timing Tool Procedures
%%Creator: RH
%%CreationDate: July 19, 1991
%%Copyright: 1991 by Adobe Systems Incorporated
%%EndComments
%
% Copyright (C) 1991 by Adobe Systems Incorporated.
% All rights reserved.
%
%%BeginExitServer: 0
serverdict begin 0 exitserver

/TIMETOOL (incomplete) def
/elapsedtime 0 def
/timeToolDict 20 dict def

systemdict /realtime known
   { /timerop (using realtime: ) def /gettime /realtime load def }
  { /timerop (using usertime: ) def /gettime /usertime load def }
  ifelse

/beginTiming {  %  -  beginTiming  -    begin interval timing
   /tmptime gettime def
} bind def
/endTiming {  %  -  endTiming  -    end interval timing
   gettime tmptime sub 1000 div /elapsedtime exch def
} bind def

/TIMETOOL (interval) def
```

```
%!PS-Adobe-3.0 ExitServer
%%Title: Download Report To Screen Timing Tool Procedure
%%Creator: RH
%%CreationDate: August 14, 1991
%%Copyright: 1991 by Adobe Systems Incorporated
%%EndComments
%
% Copyright (C) by Adobe Systems Incorporated.
% All rights reserved.
%
%%BeginExitServer: 0
serverdict begin 0 exitserver
%%BeginProlog

/reportTime { %  -  reportTime  -  report value of elapsed time to screen
  (Time obtained ) print timerop print
  elapsedtime 6 string cvs print ( seconds) print
} bind def
```

For more help:
(520) 428-4073

Adobe timing utilities...

```
%!PS-Adobe-3.0 ExitServer
%%Title: Download Report On Page Timing Tool Procedure
%%Creator: RH
%%CreationDate: August 14, 1991
%%Copyright: 1991 by Adobe Systems Incorporated
%%EndComments
%
% Copyright (C) by Adobe Systems Incorporated.
% All rights reserved.
%
%%BeginExitServer: 0
serverdict begin 0 exitserver

/reportTime { % - reportTime - report value of elapsed time on page
  save
  0 0 moveto 0 72 rlineto 288 0 rlineto 0 -72 rlineto closepath
  1 setgray fill % make sure report will be visible
  /Times-Roman findfont 8 scalefont setfont 0 setgray
  40 56 moveto currentpoint % initial line position
  statusdict begin
    /product where { pop product show ( ) show } if
    /printername where
    { pop ( known as ) show 64 string printername show } if
  end
  moveto 8 -10 rmoveto currentpoint % next line
  /languagelevel where { pop (PostScript Level 2) show }
            { (PostScript Level 1 ) show }
            ifelse
  (  version ) show version show
  moveto 8 -10 rmoveto currentpoint % next line
  (Name of job is ) show
  statusdict begin
    /jobname where { pop jobname type /stringtype eq
             { jobname }
             { (not given) } ifelse }
            { (not given) } ifelse show
  end
  moveto 8 -10 rmoveto  % last line
  (elapsed time measured ) show timerop show
  elapsedtime 10 string cvs show ( seconds) show
  restore
} bind def
```

```
%!Copyright 1985,1986 Adobe Systems Inc.  All Rights Reserved.
% timepages.ps
```

Sending this at the beginning of any PostScript job will cause the execution time (exclusive of printing time) of each page to be printed at the top of all pages as follows:

<productname> <version#> execution time = <x> + <y> = <total> seconds

where <x> is execution time up to "showpage" (including waiting for i/o) and <y> is execution time (if any) for low level graphics operators executed after "showpage".

```
/origshowpage /showpage load def

/showpage
   {userdict begin
    /startdump usertime def
    dumpdisplaylist
    /enddump usertime def
    /starttime decodetime def
```

Don Lancaster's
PostScript
Secrets

Bonus
Supplement
#28-B

Adobe timing
utilities

For more help:
(520) 428-4073

Bonus Supplement #28-C

Adobe timing utilities

Adobe timing utilities...

```
    /exectime startdump starttime sub 1000 div def
    /dumptime enddump startdump sub dumpoverhead sub 1000 div def
    dumptime .004 lt {/dumptime 0.0 def} if
    gsave
    initgraphics
    72 10.55 72 mul moveto 0 16 rlineto 6.5 72 mul 0 rlineto 0 -16 rlineto
    closepath 1 setgray fill
    0 setgray
    /Times-Roman findfont 12 scalefont setfont
    100 10.6 72 mul moveto
    statusdict begin product end show (   ) show version show
    (  execution time = ) show
    exectime =string cvs show ( + ) show
    dumptime =string cvs show ( = ) show
    exectime dumptime add =string cvs show ( seconds) show
    grestore
    origshowpage
    statusdict /redclose known {statusdict /redclose get exec} if
    statusdict /printerclose known {statusdict /printerclose get exec} if
    statusdict /scoutclose known {statusdict /scoutclose get exec} if
    statusdict /atlasclose known {statusdict /atlasclose get exec} if
    usertime encodetime
    end
   } def

/dumpdisplaylist
   {gsave initgraphics
    100 500 translate
    1 1 8 [1 0 0 1 0 0]  {(ÿ)} image
    grestore
   } def

/dumpoverhead usertime dumpdisplaylist usertime exch sub def

/encodetime
   {3 -1 0
       {startstr exch 2 index 16#FF and put
  -8 bitshift
       } for
    pop
   } def

/decodetime
   {0 startstr {exch 8 bitshift add} forall} def

/startstr 4 string def
usertime encodetime

gsave
/Times-Roman findfont 12 scalefont setfont
100 10 72 mul moveto
(This page inserted by timepages.ps) show
grestore
showpage
```

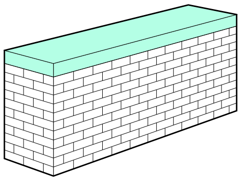
Don Lancaster's PostScript Secrets

29

As a reminder, most applications packages always seem to insist on using the *seventeenth* most putrid LaserWriter gray. Here's an infinitely better gray . . .

```
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen
```

Other interesting combinations are **75 15** for use on halftone photos, **85 35** for a reduced reproduction, or **133 25** for india ink wash effects. See the Bonus Supplement #18 for one of my infamous secret gray maps.



The very next time that you are up against the wall, an infuriatingly subtle misuse of the **setdash** operator will ridiculously speed up drawing all of your perspective bricks. Each dash is set one brick high when mapped into the 3-D space. A listing is on the next page.

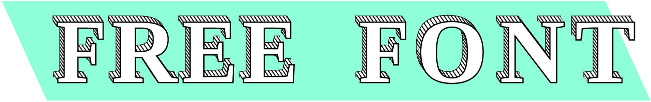
Here's more rules to minimize NTX hard disk blowups: Use a minimum of 3 Megabytes of RAM. Stay 100% host backed up. Always turn the disk on first, followed by the printer, and vice versa. Issue a **[control]-c** every now and then and don't ever terminate a disk cache update. Do a cold reboot at least three times a day. Re-initialize once per month. Never send anything new to your printer while the disk light is active. Issue a **[control]-c** before shutdown.

MATERIAL OF THE MONTH —

Low cost **THERMAL BINDING TAPES** are available from *Planax* and can use most any cover materials of your choice. Unfortunately, their binding machine pricing is ludicrously absurd, so you will want to whip up your own $49.95 homebrew substitute. *Planax North America*, 15 East 26th Street, Suite 1908, New York, NY, 10010. (212) 532-1988.

PUBLICATIONS OF THE MONTH —

By far the two most important PostScript books are the **BLUE BOOK**, otherwise known as the *PostScript Tutorial and Cookbook* and the **RED BOOK**, also called the *PostScript Reference Manual.* One stocking source is *Synergetics*, Box 809, Thatcher AZ, 85552. (602) 428-4073.



Here is yet another subtle misuse of the **setdash** operator, buried in a universal 3-D font routine . . .

```
/charproc {outlineweight setlinewidth gsave depth shadowangle cos mul
depth shadowangle sin mul translate charstring newpath 0 0 moveto
charstring false charpath stroke depthproc depth 2 mul {newpath 0 0
moveto charstring false charpath gsave 1 setgray fill grestore stroke 0.5
shadowangle cos mul neg 0.5 shadowangle sin mul neg translate} repeat
grestore charstring false charpath gsave 1 setgray fill grestore stroke} def
/charstring (X) def

/3dfont { gsave {charstring exch  0 exch put 0 0 moveto charstring charproc
charstring stringwidth pop extrakern add 0 translate} forall grestore} def

% demo - remove before use.

/Palatino-Bold findfont [50 0 0 50 0 0] makefont setfont /depth 4 def
/outlineweight 0.7 def /depthproc {[0.5 1.5] 0.5 setdash} def /extrakern 3
def /shadowangle 135 def

100 200 translate -5 -10 moveto -55 55 rlineto 350 0 rlineto 60 -60 rlineto
closepath bestgray 0.85 setgray fill 0 setgray (FREE  FONT) 3dfont
showpage quit % bestgray = 106 45 setscreen per above
```

For more help:
(602) 428-4073

Don Lancaster's LaserWriter Secrets

Bonus Supplement #29

When You Are Up Against the Wall

A fast 3-D perspective brick routine...

Here are some excerpts from my perspective drawing routines found in my *PostScript Show & Tell...*

```
/px {/zz exch def /yy exch def /xx exch def yo dup yy add div dup xx xo sub
mul exch zz zo sub mul} def /psave {/zh zz def /yh yy def /xh xx def} def /pm
{px psave moveto} def

/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 {/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

/xrd {0 0 prd} def /yrd {0 exch 0 prd} def /zrd {0 exch 0 exch prd} def /zrm {0
exch 0 exch prm} def

/xzrect {/zdis exch def /xdis exch def save zdis zrd xdis xrd zdis neg zrd
closepath gsave currentshade setgray fill grestore stroke restore} def

/yzrect {/zdis exch def /ydis exch def save zdis zrd ydis yrd zdis neg zrd
closepath gsave currentshade setgray fill grestore stroke restore} def

/pbox {/zdis exch def /ydis exch def /xdis exch def save xdis zdis xzrect
restore save ydis zdis yzrect restore save zdis zrm xdis ydis xyrect restore}
def

/xyrect {/ydis exch def /xdis exch def save ydis yrd xdis xrd ydis
neg yrd closepath gsave currentshade setgray fill grestore stroke
restore} def
```

And here are two of the magic brick routines. The *setdash* operator is set on the fly to one brick high in the perspective space...

```
/xzbrickwall {gsave /zbri exch def /xbri exch def xbri brickwide mul 0 0 prm
0 1 zbri 1 sub cvi {gsave pop xbri brickwide mul neg 0 brickhi prm xbri
brickwide mul  0 0 prd stroke grestore} for xbri brickwide neg mul 0 0 prm 0
1 xbri 1 sub cvi {gsave pop brickwide 0 zbri brickhi mul neg prm
currentpoint exch pop /cph exch def 0 0 zbri brickhi mul prd cph
currentpoint exch pop sub abs zbri div mark exch ] 0 setdash stroke
grestore} for xbri  brickwide neg mul 0 0 prm 0 1 xbri 1 sub cvi {gsave pop
brickwide 0 zbri brickhi mul neg prm currentpoint exch pop /cph exch def 0
0 zbri brickhi mul prd cph currentpoint exch pop sub abs zbri div dup mark
exch ] exch setdash stroke grestore } for} def

/yzbrickwall { gsave /zbri exch def /ybri exch def 0 ybri brickwide mul 0 prm
0 1 zbri 1 sub cvi {gsave pop 0 ybri brickwide mul neg brickhi prm 0 ybri
brickwide mul 0 prd stroke grestore} for 0 ybri brickwide neg mul 0 prm 0 1
ybri 1 sub cvi {gsave pop 0 brickwide zbri brickhi mul neg prm currentpoint
exch pop /cph exch def 0 0 zbri brickhi mul prd cph currentpoint exch pop
sub abs zbri div dup mark exch ] exch setdash stroke grestore } for 0 ybri
brickwide neg mul 0 prm 0 1 ybri 1 sub cvi {gsave pop 0 brickwide zbri
brickhi mul neg prm currentpoint exch pop /cph exch def 0 0 zbri brickhi
mul prd cph currentpoint exch pop sub abs zbri div mark exch ] 0 setdash
stroke grestore } for} def
```

Here is how it all goes together...

```
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen

160 710 translate 0 setlinewidth /xo 00 def /yo 700 def /zo 250 def /objrot
45 def /brickhi 5 def /brickwide 16 def /numbricksx 10.5 def /numbricksy
2.5 def /numbricksz 12.5 def /bldz numbricksz brickhi mul def /bldx
numbricksx brickwide mul def/bldy numbricksy brickwide mul def

/currentshade 1 def 0 0 0 pm bldx bldy bldz pbox /currentshade 0.9 def 0 0
brickhi 11 mul pm  bldx bldy brickhi 1.5 mul pbox /currentshade 1 def

0 0 0 pm numbricksx numbricksz 1.5 sub xzbrickwall
0 0 0 pm numbricksy numbricksz 1.5 sub yzbrickwall

0 0 0 pm  numbricksy brickwide mul yrd numbricksz brickhi mul zrd
numbricksx brickwide mul xrd numbricksy brickwide mul neg yrd
numbricksz brickhi mul neg zrd closepath 0.8 setlinewidth stroke

showpage quit
```

See the previous page for a sample brickwall graphic printout.

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

30

Paper curling problems can be eased with several of these tricks: Keep the humidity between 20 and 38 percent, using a dehumidifier if necessary; Use your "straight through" paper path on the NT or NTX; Keep paper wrapped and flat on steel shelves until one day before use, then open and acclimatize for 24 hours; Try pressure clamping or refrigeration; Replace the NT or NTX pickup rollers and pads every 25,000 copies. But short grain paper is a no-no.

Here's some uses for the PostScript random number generator...

```
rand 32768 div 65536 div                       % for fp 0.0000 to 0.9999
k rand 32768 div 65536 div mul floor           % for integer 0 to k-1
k rand 32768 div 65536 div mul floor 1 add     % for integer 1 to k
rand -24 bitshift                              % for integer 0 to 127
```

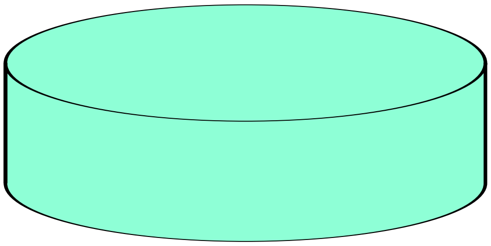
But note that the faster **rand 127 and** does *not* work. Its very short sequence suggests something flakey about Adobe's alogrithm. Why is this so?

MATERIAL OF THE MONTH —

LaserWriter compatible **RUBBER STAMP SUPPLIES** are now available retail from *Grantham's* at (218) 773-0331, in moderate quantites through *R. A. Stewart* at (213) 690-4445, or from the *Merigraph* manufacturer at (312) 961-4065.

PUBLICATION OF THE MONTH —

The free **ALPHABET SOUPS** typeface directory that is both a good typography source book and an excellent soup cookbook. Be certain to try the Roquefort soup. From *International Typeface Corporation* at (212) 371-0699.



Here's some canned PostScript font layout code that you just may want to wrap yourself around...

```
/pixellineremap {0 1 pxpwidth 300 mul 72 div cvi {/sline# exch def save
/snap1 exch def gsave mappingproc newpath sline# 72 mul 300 div 0
moveto 0 pxpheight rlineto 0 0 rlineto 0 pxpheight neg rlineto closepath
clip newpath pixelproc grestore clear snap1 restore} for } def

/mappingproc {sline# .24 mul dup neg exch pxpwidth 2 div sub canwide div
114.6 mul dup sin canwide 2 div mul exch 3 1 roll add exch cos 1 exch sub
tiltangle sin canwide 2 div mul mul translate} def

% demo - remove before use.

106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen
200 300 translate

/tiltangle 20 def /canhi 50 tiltangle sin div def /canwide 200 def
/NewCenturySchlbk-Bold findfont [30 0 0 33 0 0] makefont setfont
/labelypos 8 def /pxpwidth 238 def /pxpheight 35 def

gsave newpath 1 dup tiltangle sin scale 1.5 setlinewidth 0 canhi canwide 2
div 0 180 arc 0 0 canwide 2 div 180 0 arc canwide 2 div neg canhi moveto 0
canhi neg rlineto canwide 2 div 0 moveto 0 canhi rlineto gsave 0.85 setgray
fill grestore 0 canhi canwide 2 div 0 180 arcn stroke grestore 0 canwide 2
div tiltangle sin mul neg labelypos add translate % draws the can

/pixelproc { 0 0 moveto 0 pxpheight rlineto pxpwidth 0 rlineto 0 pxpheight
neg rlineto closepath gsave 1 setgray fill grestore 4 setlinewidth stroke 12
6 moveto 1 0 (FREE  FONT) ashow} def    % this is the flat label

pixellineremap showpage quit % pastes label on can
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

31

Don't forget that the green knob is backwards on the NT and the NTX. The "9" setting uses the *least* toner! Always use the *highest* number that gives you an acceptable print quality. Watch this detail.

Any unbound PostScript operator could get diverted for nefarious uses. Here's how you can redefine **showpage** so it can auto-print a letterhead, border, 3-hole punch marks, or other message...

```
100 dict /workdict exch def workdict begin % MUST be out of userdict!

/printtimestuff {save /pts exch def mark /Helvetica-Bold findfont [80 0 0 80
0 0] makefont setfont /msg (FILE  COPY) def 0 setlinewidth 1 setlinejoin 1
setlinecap 170 240 translate  45 rotate 0 0 moveto /strrx (X) def msg {strrx
exch 0 exch put strrx true charpath currentpoint stroke newpath moveto}
forall cleartomark pts restore} def

/showpage {printtimestuff systemdict /showpage get cvx exec} def

% All of your normal page stuff goes here

showpage quit
```

Note that **workdict** must be *above* **userdict** on your dictionary stack for this to work. All **showpage** commands must also be unbound.

One way to speed up printing a custom logo is to convert it into a character in a font and then cache your font character. The font bitmap can be returned from the NTX hard disk to the host for a further dramatic speedup.

MATERIAL OF THE MONTH —

LaserWriter compatible **THERMAL TRANSFER TONER** which can iron directly onto T-shirts or other fabrics in four colors is now available from *Black Lightning* at (800) 252-2599, or *Lazer Products* at (303) 792-5277, or *Don Thompson* at (714) 855-3838. Black Lightning has free samples. The stuff is also ideal for prototype printed circuits or printing on beer cans.

PUBLICATIONS OF THE MONTH —

One each of everything is available for $189 yearly as a portion of the **ADOBE DEVELOPER'S PROGRAM**. Contact Cynthia Johnson at *Adobe Systems Inc.*, 1585 Charleston Road, Mountain View, CA 94039. (415) 961-4400.

FREE FONT

And you thought Palatino was snotty. Some tricks here include an *erosion* by white outlining and repeat clipped stroking...

```
/superinsidestroke {save clip /sssnap exch def /sscmd exch def mark 0 2
sscmd length 2 div cvi 1 sub 2 mul {/aposn exch def gsave sscmd aposn get
2 mul setlinewidth sscmd aposn 1 add get setgray stroke grestore} for
cleartomark sssnap restore newpath} def

/showitstuff {currentpoint [4 0  3 1 1.7 0 0.7 1] superinsidestroke exch
kernstuff add exch moveto} def
/strrx (X) def /kernstuff -1 def

/snottyfont{ save /snotsnap exch def /msg exch def translate mark 0 0
moveto msg {strrx exch 0 exch put strrx false charpath showitstuff initclip}
forall cleartomark snotsnap restore} def

% demo - remove before use.

/Palatino-Bold findfont [50 0 0 50 0 0] makefont setfont
200 300 (FREE  FONT) snottyfont showpage quit
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

32

A true radial font

Die-O-Perf products

Drawing a "no" logo

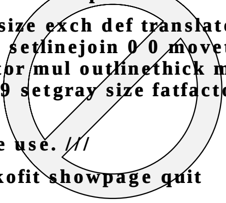
Parchment path tool

Recharger newsletter

A 4 x 11 inch piece of parchment cover stock makes a good tool for easing any paper feeding problems. Vigorously saw the sheet back and forth, paying extra attention to the paper path edges.

Exactly which part of *no* didn't you understand? Like so...

```
/bestgray {106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen} def

/donteventhinkofit {gsave /size exch def translate -45 rotate /fatfactor 0.12
def /outlinethick 1.10 def 2 setlinejoin 0 0 moveto 0 0 size 2 div 0 540 arc
closepath gsave size fatfactor mul outlinethick mul setlinewidth 0 setgray
stroke grestore bestgray 0.9 setgray size fatfactor mul setlinewidth stroke
grestore } def

% /// demo - remove before use. ///

200 300 200 donteventhinkofit showpage quit
```



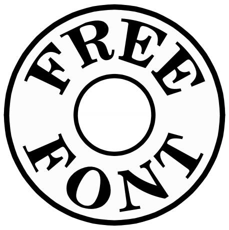
Squiggly brackets will *defer* the execution of PostScript until your routine gets called. A **/zowie gleek zounds add def** takes **gleek** and adds it to **zounds** at the definition time. A **/zowie {gleek zounds add} def** takes **gleek** and adds it to **zounds**, at the time it gets called.

MATERIAL OF THE MONTH —

Laser printable **DIE CUT BLANKS** for tickets, tents, tags, response mailers, coupons, and hangers. Around a nickel per sheet in eight colors. They feed and take toner beautifully. Through *Die-O-Perf* at 1721 East Pioneer Drive, Irving, TX, 75061. (800) 843-2807.

PUBLICATIONS OF THE MONTH —

RECHARGER is a newsletter that covers toner refilling. While their first issue is less than spectacular, free samples are available, and it just might work into something. $25 per year from *Recharger*, 3870 LaSierra Avenue, Suite 266, in Riverside, CA, 92505. (714) 359-8570.



This true radial font has curved tops and bottoms and gets fatter the further it strays from dead center. Here's one possibility...

```
/vpixellineremap {0 1 oversample mul pxpwidth 4.166667 mul cvi {/sline#
exch def save /snap1 exch def gsave mappingproc newpath sline# 72 mul
300 div 0  moveto 0 pxpheight rlineto 0 0 rlineto 0 pxpheight neg rlineto
closepath clip newpath pixelproc grestore clear snap1 restore} for } bind def
/bottomhalf false def

/mappingproc {sline# 44 pxpwidth div mul neg 91 add bottomhalf {neg} if
rotate sline# -0.24 mul bottomhalf {pxpheight neg 1.333 mul}{0} ifelse
translate} bind def

/pixelproc {6 setlinewidth 0 50 moveto 100 0 rlineto stroke 3 setlinewidth 0
18 moveto 100 0 rlineto stroke 3 25 moveto 1 0 bottomhalf
{botmsg}{topmsg} ifelse ashow} def

% /// demo - remove or alter before reuse. ///

mark 12 12 setcacheparams /NewCenturySchlbk-Bold findfont [25 0 0 25 0
0] makefont setfont /pxpwidth 100 def /pxpheight 50 def /oversample 0.60
def /topmsg ( FREE) def /botmsg ( FONT) def

200 400 translate /bottomhalf false def vpixellineremap /bottomhalf true
def vpixellineremap showpage quit
```

For more help:

(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #32-A

Spherical nonlinear transform

Spherical nonlinear transformations...

This code is the first in a new series of unique nonlinear transformations that allow you to rapidly and conveniently draw on apparently nonflat surfaces. This particular example creates a globe or a sphere, given text and graphics specified as latitude and longitude.

The code is presented in several sections. The first is a generalized new nonlinear transformer, intended primarily for PS level II use. The second includes spherical drawing utilities to draw the outline, and latitude and longitude lines. The third is a compiled example of how captured font paths can be shown on level I machines.

Remember the following restrictions when using level I: The maximum number of elements on the stack is around 500; the maximum path length is around 1500; and the #%$@$# font lockout generates an error message if you try to use charpath followed by pathforall. Note also that the distillery traps out charpaths.

```
%  BE SURE TO CLEAR THIS FLAG IF YOU DO NOT HAVE LEVEL II AVAILABLE!

/usinglevel2 false def
```

GENERALIZED GURU NONLINEAR TRANSFORMER

A general nonlinear transformation allows you to redefine moveto, lineto, curvetrace, and closepath to map them onto any desired surface.

This transformer works best with PostScript level 2, where unlocked font paths are instantly available. It may be used with PS level 1 if no font paths are used, or if font paths are predefined.

It is also possible to use PS level 2 once, and then return the nonlinear path for host recording. After recording and a path length check, the results should run on any level machine.

Since some nonlinear transformations will convert straight lines into curved lines, you might have to autoconvert longer straight lines into sequential multiple curves to prevent "short cuts".

The maxstraight variable determines how long the longest line is allowed to be without changing it into two or more curvea. This prevents "short cuts". A very large value turns this feature off..

```
/maxstraight 5000 def  % maximum allowable straight line length
```

Because of the need to prevent shortcuts, "currentpoints" in the PRE-transformed space must be preserved. The last currentpoint x0 and y0 for lineto to curveto conversions, and the initial moveto currentpoint for closepaths.

```
/savecp {2 copy /y0 exch def /x0 exch def} def

/mt {savecp /xx0 x0 def /yy0 y0 def nlt moveto} def
/li {noshortcuts} def
/ct {savecp 3 {6 2 roll nlt} repeat curveto} def
/cp {xx0 yy0 li closepath} def

% nlmap does a nonlinear mapping of an ALREADY CREATED PATH using an nlt
% algorithm to do the actual mapping. It produces an on-stack array of the
% nonlinear mapping...

/nlmap {gsave mark /newpath cvx {/mt cvx}{/li cvx}{/ct cvx}
{/cp cvx} pathforall newpath] grestore cvx} def
```

For more help:
(520) 428-4073

Spherical nonlinear transformations...

```
% makecurves converts long lines into Bezier curvetos with control points on the
line at the .333 and .667 points. This approximation is simple and fairly good
looking. It is easily improved by reducing maxstraight at the expense of longer
files and runtimes.

/makecurves {floor cvi y1 y0 sub 1 index div 3 div /ydelta
exch def x1 x0 sub 1 index div 3 div /xdelta exch def {x0 y0 ydelta
add exch xdelta add exch 2 copy ydelta add exch xdelta add exch
2 copy ydelta add exch xdelta add exch ct} repeat} def

% noshortcuts decides whether a lineto is long enough to need conversion into
two or more curvetos...

/noshortcuts {/y1 exch def /x1 exch def y1 y0 sub abs x1 x0 sub abs
add maxstraight div dup 1 gt {makecurves}{pop x1 y1 savecp nlt lineto}
ifelse} def
```

SPECIFIC GURU SPHERICAL TRANSFORMER

The spherical transform is deceptively simple. Assume a sphere of unit radius, with latitude ranging from -90 degrees (south pole) to +90 degrees (north pole) and longitude ranging from -90 degrees (far west) to +90 (far east). Assume each input point is defined by its latitude and longitude in decimal degrees.

the transform is simply...

```
    x’ = cos (latitude) * sin (longitude)
    y’ = sin (latitude)
```

done with this operator....

```
     /spxf {dup sin 3 1 roll cos exch sin mul exch} def
```

A slightly fancier (and somewhat slower) transform could be used to clip anything greater than +90 or -90 degrees to make anything on the back side invisible.

```
% nlt is the link to the nonlinear transform currently in use....

/nlt {spxf} def
```

And the maximum allowable straight line is around five degrees or so. Use less for more accuracy; more for faster execution...

```
/maxstraight 5 def  % maximum allowable straight line length
```

SPHERE DRAWING UTILITIES

```
% latlon draws latitude and longitude curves. The method shown is level I
friendly. For level 2, just piling the entire path up on the stack without any
saves/restores is faster and more flexible.

/latres 18 def  % latitude resolution in degrees
/lonres 18 def  % longitude resolution in degrees

/latlon {newpath -90 lonres 90 {save /llsnap exch def -90 moveto 0 180 rlineto
nlmap exec stroke llsnap restore} for -90 latres 90 {save /llsnap exch def  -90 exch
moveto 180 0 rlineto nlmap exec stroke} for llsnap restore} def

% spoutline emphasizes the sphere outline

/spoutline {-90 -90 moveto 180 0 rlineto 0 180 rlineto -180 0 rlineto closepath}
def

% cspheremsg centers a character message on x y (msg).

/cspheremsg {/msg exch def /ypos exch def /xpos exch def newpath 0 0 moveto
xpos msg stringwidth pop 2 div sub ypos moveto msg false charpath} def
```

Bonus Supplement #32-B

Spherical
nonlinear transform

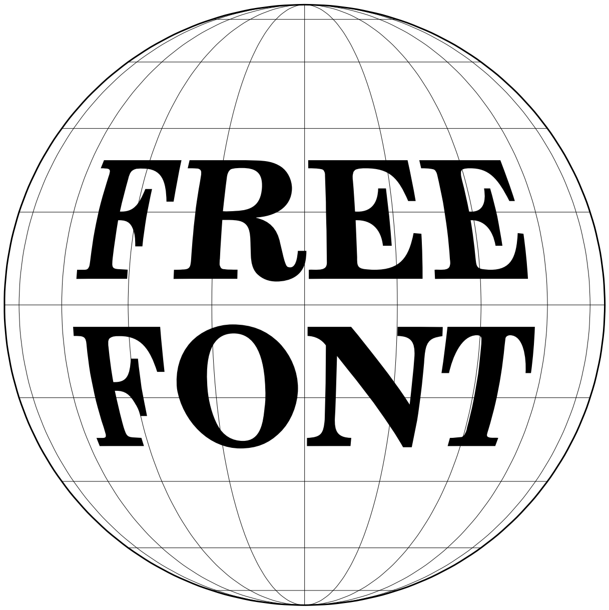
Don Lancaster's PostScript Secrets

Bonus Supplement #32-C

Spherical
nonlinear transform

Spherical nonlinear transformations...

```
%  //// demo - remove or alter before reuse ////

%  use save and restore to minimize vm usage
save /spheresnap exch def

% pick a location and scale the unit radius sphere...
200 200 translate 100 dup scale

% draw the latitude and longitude lines...
gsave 0 setlinewidth latlon grestore

% make the outline slightly heavier...
gsave spoutline nlmap exec .005 setlinewidth stroke grestore

% put some words on the sphere. Font size is in degrees..
/NewCenturySchlbk-Bold findfont [33 0 0 33 0 0] makefont setfont

usinglevel2 {
gsave 0   5 (FREE) cspheremsg nlmap cvx exec fill grestore
gsave 0 -28 (FONT) cspheremsg nlmap cvx exec fill grestore
} if

spheresnap restore   % recycle vm
```



Note: See companion disk for host recorded code runtime sample for level 1 and GhostScript.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

33

Static cling vinyl
Large letters font
I/O error messages
Adobe Type Manager
Font lists & printouts
Thermal binding ideas

A LaserWriter *I/O ERROR* message after several pages have gotten printed almost certainly tells you your host did not properly activate its *XON/XOFF* handshaking. Your input buffer usually holds 6000 characters. Failure to do your handshake will overflow this buffer.

The *exact* spelling of any font is super important and can become maddeningly frustrating. Here are two routines that list all of your available fonts for you...

```
/listfonts {FontDirectory {pop == flush 200 {37 sin pop} repeat} forall} def
%send installed font list to host

/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 % on paper
```

You might want to add these to your nuisance command dictionary. A list of commonly used fonts appears on the next page.

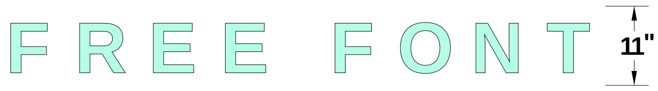
Here's three tips that can improve the reliability of all your thermal bindings in a big way: (1) Always jog all your sheets; (2) Notch the spine area of the stack with five or six fairly deep groves; and (3) Preheat your pages before binding.

MATERIAL OF THE MONTH —

LaserWriter printable **STATIC CLING VINYL** and your X-Acto knife will convert your LaserWriter into a $15,000 signmaking machine. About 55 cents per cut sheet in five colors, this will easily and semi-permanently stick to glass and other smooth surfaces. Do be certain to trim 3/8 of an inch of vinyl off the backing sheet edges before hand feeding. From *Joseph Struhl Company*, at 95 Atlantic Avenue, Garden City Park, NY 11040, (800) 552-0023.

PUBLICATION OF THE MONTH —

The **ADOBE TYPE MANAGER** instantly and transparently prints full quality PostScript fonts on your Mac screen, ImageWriter, or on any other printer with superb quality results. $99 from *Adobe Systems*, at 1585 Charleston Road, Mountain View, CA 94039. (415) 961-4400. One discounting source would be *MacWarehouse* at (800) 255-6227.



This free font will convert a message into individual giant letter patterns for use with static cling vinyl, stencils, homecoming floats, storefront signage, or just for big house numbers...

```
/inches {72 mul} def /str (X) def

/makebigletters {fonttouse findfont [letterheight 0 0 letterwidth 0 0]
makefont setfont {str exch 0 exch put str dup stringwidth pop 8.5 72 mul
exch sub 2 div 11 72 mul letterheight sub moveto false charpath stroke
showpage} forall} def

% //// demo - remove before use ////

/letterheight 9 inches def   % From top of "W" to bottom of "g".
/letterwidth 9 inches def    % Usually same as letterheight.

/fonttouse {/NewCenturySchlbk-Bold} def   % watch the spelling!

(FREE FONT) makebigletters

quit %  no showpage needed
```

You can easily extend the routine to show the sidebearings, or for letters so big they take several sheets to print. Several smaller letters could also be shown on the same sheet, as could a stencil library of the full alphabet.

For more help:
(602) 428-4073

Don Lancaster's LaserWriter Secrets

Bonus Supplement #33

A list of the often-used fonts

Correct spellings for some common fonts...

Be sure to use these spellings when picking your fonts ...

/Aachen-Bold
/AvantGarde-Book
/AvantGarde-BookOblique
/AvantGarde-Demi
/AvantGarde-DemiOblique

/BernhardFashion
/Benguiat-Bold
/Benguiat-Book
/Bodoni
/Bodoni-Bold
/Bodoni-BoldItalic
/Bodoni-Italic
/Bodoni-Poster
/Bookman-Demi
/Bookman-DemiItalic
/Bookman-Light
/Bookman-LightItalic
/BrushScript

/Carta
/CenturyOldStyle-Regular
/CenturyOldStyle-Bold
/CenturyOldStyle-Italic
/Classic
/ClelaBor1805
/CooperBlack
/CooperBlack-Italic
/Courier
/Courier-Bold
/Courier-BoldOblique
/Courier-Oblique

/Eurostile-Condensed
/Eurostile-ExtendedTwo
/Eurostile-BoldExtendedTwo

/FreestyleScript
/FrizQuadrata
/FrizQuadrata-Bold

/Garamond-Bold
/Garamond-BoldItalic
/Garamond-Light
/Garamond-LightItalic
/Goudy
/Goudy-Bold
/Goudy-BoldItalic
/Goudy-Italic

/Helvetica
/Helvetica-Black
/Helvetica-BlackOblique
/Helvetica-Bold
/Helvetica-BoldOblique
/Helvetica-Condensed
/Helvetica-CondensedBold
/Helvetica-CondensedBoldOblique
/Helvetica-CondensedOblique
/Helvetica-Light
/Helvetica-LightOblique
/Helvetica-Narrow
/Helvetica-Narrow-Bold
/Helvetica-Narrow-BoldOblique
/Helvetica-Narrow-Oblique
/Helvetica-Oblique
/Hobo

/LubalinGraph-Book
/LubalinGraph-BookOblique
/LubalinGraph-Demi
/LubalinGraph-DemiOblique
/Lucida
/Lucida-Bold
/Lucida-BoldItalic
/Lucida-Italic

/Melior
/Melior-Bold
/Melior-BoldItalic
/Melior-Italic

/NewBaskerville-Bold
/NewBaskerville-BoldItalic
/NewBaskerville-Italic
/NewBaskerville-Roman
/NewCenturySchlbk-Bold
/NewCenturySchlbk-BoldItalic
/NewCenturySchlbk-Italic
/NewCenturySchlbk-Roman

/Optima
/Optima-Bold
/Optima-BoldOblique
/Optima-Oblique

/Palatino-Bold
/Palatino-BoldItalic
/Palatino-Italic
/Palatino-Roman
/ParkAvenue

/QuillscriptFlowers

/Revue
/Round

/Showtime
/Sonata
/Souvenir-Demi
/Souvenir-DemiItalic
/Souvenir-Light
/Souvenir-LightItalic
/Stencil
/StoneSans
/StoneSans-Bold
/StoneSans-BoldItalic
/StoneSans-Italic
/StoneSans-Semibold
/StoneSans-SemiboldItalic
/Symbol

/ThompsonQuillscript
/Times-Bold
/Times-BoldItalic
/Times-Italic
/Times-Roman
/TrumpMediaeval-Bold
/TrumpMediaeval-BoldItalic
/TrumpMediaeval-Italic
/TrumpMediaeval-Roman

/UniversityRoman

/WeddingText

/ZapfDingbats
/ZapfChancery-MediumItalic

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

34

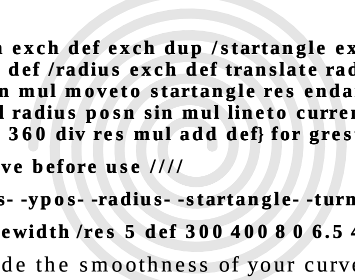
Autoboxing font
PostScript spiral
Exclusive-or procs
Menu cover sources
A *Printer's Ink* book
Persistent downloads

Another way to express the logical **exclusive-or** is **or but not and**. Here's one method to handle PostScript's missing **eor** operator...

```
a b or a b and not and  % leaves a eor b on stack
```

This routine is handy to generate simple spirals and several other rather useful art effects...

```
/spiral {gsave /pitch exch def exch dup /startangle exch def 360 div add 360
mul /endangle exch def /radius exch def translate radius startangle cos mul
radius startangle sin mul moveto startangle res endangle {/posn exch def
radius posn cos mul radius posn sin mul lineto currentpoint stroke moveto
/radius radius pitch 360 div res mul add def} for grestore} def

% //// demo - remove before use ////

% sequence is -xpos- -ypos- -radius- -startangle- -turns- -pitch-

1 setlinecap 2 setlinewidth /res 5 def 300 400 8 0 6.5 4 spiral showpage quit
```



The **res** value does trade the smoothness of your curve against your processing time. For some real surprises, try using **res** values of 60 and 144.

Note that a persistently downloaded font accesses a few seconds faster than one picked up each time off your NTX hard disk. To save time while book-on-demand printing page files, persistently download major fonts before starting.

MATERIALS OF THE MONTH —

Sources of **MENU COVERS** include *Printer's Shopper* at (800) 854-2911, *H. Risch* located at (716) 232-6938, *Menu Graphics* found at (800) 321-9482, *Menu Cover* at (716) 856-5430, and *MenuTech* found at (800) 535-MENU.

PUBLICATION OF THE MONTH —

A free hardback book titled **PRINTERS INK: A SELECTION** gives you bunches of real insider secrets to smaller quantity book publishing. From *Thomson-Shore*, Box 305, Dexter, MI, 48130. (313) 426-3939.

FREE FONT

Here's a free headliner font that automatically centers itself inside of a fancy and correctly-sized box for you...

```
% a non-putrid gray...
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen

/roundpath {/rpdata exch def /rprad exch def rpdata length 1 sub cvi
/rppoints exch def rpdata 0 get rpdata 1 get moveto 2 2  rppoints 2 sub
{/rpvalue exch def 0 1 3 {rpdata exch rpvalue add get } for rprad arcto pop
pop pop pop} for rpdata rppoints 1 sub get rpdata rppoints get lineto} def

/superinsidestroke {save clip /sssnap exch def /sscmd exch def 0 2 sscmd
length 2 div cvi 1 sub 2 mul {/aposn exch def gsave sscmd aposn get 2 mul
setlinewidth sscmd aposn 1 add get setgray stroke grestore} for sssnap
restore newpath} def

/abovebase 26 def /belowbase 9 def /cornerrad 8 def /extrakern 0.5 def
/strokeproc [20 .86 4 0 3 1 2 0] def

/putitinabox {gsave /msg exch def translate msg stringwidth pop msg
length 1 sub extrakern mul add 2 div /halfmsgwidth exch def cornerrad mark
0 belowbase neg halfmsgwidth neg belowbase neg halfmsgwidth neg
abovebase halfmsgwidth abovebase halfmsgwidth belowbase neg 0
belowbase neg] roundpath strokeproc superinsidestroke halfmsgwidth neg
0 moveto extrakern 0 msg ashow grestore} def

% //// demo - remove before use. ////

/NewCenturySchlbk-Bold findfont [24 0 0 24 0 0] makefont setfont 200 300
(  FREE  FONT  ) putitinabox  showpage quit
```

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #34

"Spirograph" routine

"Spirograph" style PostScript routines...

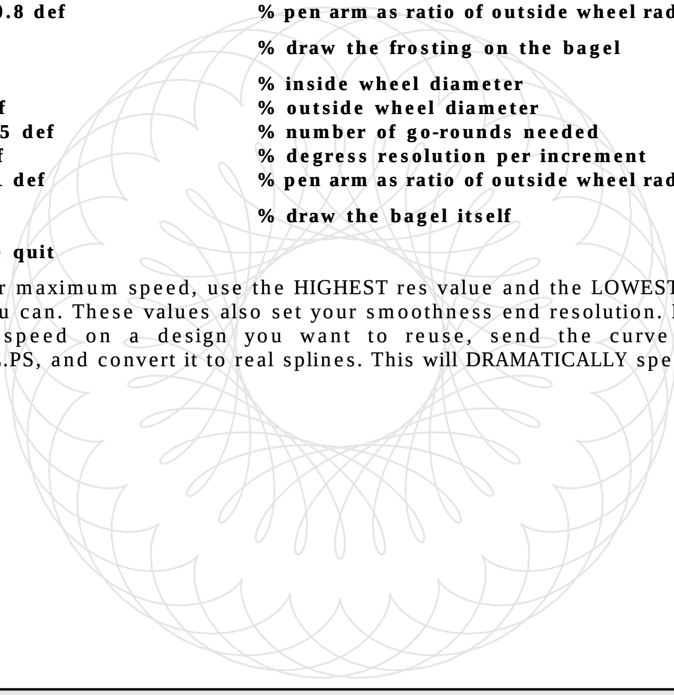
PostScript code that emulates the "super spirograph" toys. What you are generating are epocycloids and hypocycloids formed as one wheel rotates around the other. Naturally, the name "Spirograph" is a trademarked toy, and they get all bent whenever you use it.

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.

```
%  This is the magic epicycloid positioning code...

/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

%  And this routine takes care of calling it...

/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 or alter before reuse. ///

310 400 translate               % position on page
0.7 setlinewidth                % pensize
1 setlinejoin                   % round joints
1 setlinecap                    % round line caps

/a 96 def                       % inside wheel diameter
/b 75 def                       % outside wheel diameter
/maxang 25 def                  % number of go-rounds needed
/res 8 def                      % degrees resolution per increment
/penarm 0.8 def                 % pen arm as ratio of outside wheel radius

drawspiro                       % draw the frosting on the bagel

/a 96 def                       % inside wheel diameter
/b -15 def                      % outside wheel diameter
/maxang 5 def                   % number of go-rounds needed
/res 2 def                      % degress resolution per increment
/penarm 1 def                   % pen arm as ratio of outside wheel radius

drawspiro                       % draw the bagel itself

showpage quit
```

Notes: For maximum speed, use the HIGHEST res value and the LOWEST maxang values you can. These values also set your smoothness end resolution. Later, for ultimate speed on a design you want to reuse, send the curve through FUZZYBEZ.PS, and convert it to real splines. This will DRAMATICALLY speed things up.



For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

35

NT save level bug

Printer's shareware

More on *roundpath*

Starburst font ideas

Missing case command

Sandblasting & etching

The missing **case** command in PostScript is easily faked...

```
[{proc0} {proc1} ... {procn}] exch get exec  % options 0-n per stack integer
```

A zero on the stack does {proc0} and so on. Extending this to a 256 element array lets you redefine any ASCII character for any action.

My *roundpath* routine can also be used for starburst effects, just by going to a very small radius...

```
/roundpath {/rpdata exch def /rprad exch def rpdata length 1 sub cvi
/rppoints exch def rpdata 0 get rpdata 1 get moveto 2 2 rppoints 2 sub
{/rpvalue exch def 0 1 3 {rpdata exch rpvalue add get } for rprad arcto pop
pop pop pop} for rpdata rppoints 1 sub get rpdata rppoints get lineto} def

%  //// starburst demo - remove before use. ////

200 300 translate 10 dup scale 0.01 [-5 0 -7 4 -4 2 -4 6 -2 3 -1 7 1 2.5 3 6 4
3 6.5 5 6 2 8 1 5.5 -1 7.5 -3.5 5 -3 4 -6 2 -3 0 -5.5 -1 -3 -3 -6 -3 -2 -6 -5 -5 -2
-8 -2 -5 0] roundpath fill showpage quit
```

Each "point" on the burst is shown as an x,y pair in the roundpath array. To change the points on the star, first print up the *rubbergrid* of Secrets #6S.

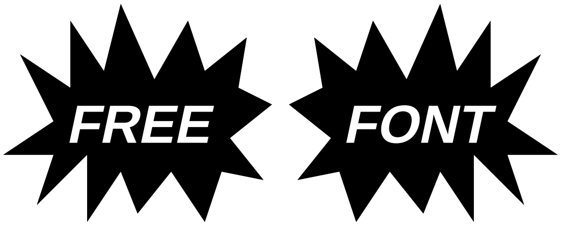
The LaserWriter NTX is swift enough to unwind things back to a much earlier save level, simply by your choosing of an earlier save object. The LaserWriter NT is not. Always do keep your saves and restores carefully paired on the NT.

MATERIAL OF THE MONTH —

Your LaserWriter can easily be used for **SANDBLASTING** and **GLASS ETCHING**. Suitable masks which you can overlay with self-stick mylar are available from *Hartco*, 1280 Glendale-Milford Road, Cincinatti, OH, 45215. (800) 543-1340. The pricing is in the $1 per square foot range.

PUBLICATION OF THE MONTH —

The free **PRINTER'S SHAREWARE** catalog now does list hundreds of shareware and public domain programs useful for printshops and desktop publishers. Most are IBM and most are very low in price. From *Printer's Shareware*, located at 5019 West Lovers Lane, in Dallas, TX 75209. (214) 350-1902.



Here's how to use the above starburst for shorter high energy or other impact messages...

```
%  requires roundpath listing as shown above

/starburst1 {gsave /msg exch def translate 0.01 [-5 0 -7 4 -4 2 -4 6 -2 3 -1 7
1 2.5 3 6 4 3 6.5 5 6 2 8 1 5.5 -1 7.5 -3.5 5 -3 4 -6 2 -3 0 -5.5 -1 -3 -3 -6 -3
-2 -6 -5 -5 -2 -8 -2 -5 0] roundpath fill msg stringwidth pop 2 div neg
currentfont /FontMatrix get 3 get 1000 mul 2.5 div neg moveto msg 1
setgray show grestore } def

% //// demo - remove before use. ////

200 300 translate 10 dup scale
/Helvetica-BoldOblique findfont [3.2 0 0 3.2 0 0] makefont setfont

10 10 (FREE) starburst1 showpage quit
```

To change the number of points or their positions, simply do a *starburst2*, etc... with different array values in *roundpath*. All of those usual 3-D and shadow effects can easily be added.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #35-A

Bezier length

Bezier curve length...

Utility and demo to find the approximate length of a single Bezier (curveto) cubic spline curve. This opens up all sorts of new design opportunities, especially when fitting text between two arbitrary curves.

WARNING: Two way comm channel required for host length reporting.

Demo to find the length along a Bezier curve and selected parametric points along a curve. Being able to use SELECTED PORTIONS of a SINGLE Bezier curve opens up all sorts of new possibilities. Not the least of which is forcing text between two arbitrary curves.

```
% Theory#1: For a parametric curve of x(t) = axt^3 + bxt^2 + cxt + x0 and
% y(t) = ayt^3 + byt^2 + cyt + y0 along with a Bezier graphing space of x0,y0,
% x1,y1, x2,y2, and x3,y3, simultaneous solution yields

%    cx = 3(x1-x0)   bx = 3x2 -6x1 +3x0   ax = x3 -3x2 +3x1 -x0
%    cy = 3(y1-y0)   by = 3y2 -6y1 +3y0   ay = y3 -3y2 +3y1 -y0

% This transforms you from graph to equation space.

% Theory#2: To approximate nearly any messy curve, break the curve up into
% straight line segments and add up their total length. The answer will always be
% a tad low, but usually converges very quickly.
```

DEMO COPY OF MY SETGRID ROUTINES from #102 UTILITY.GPS ...

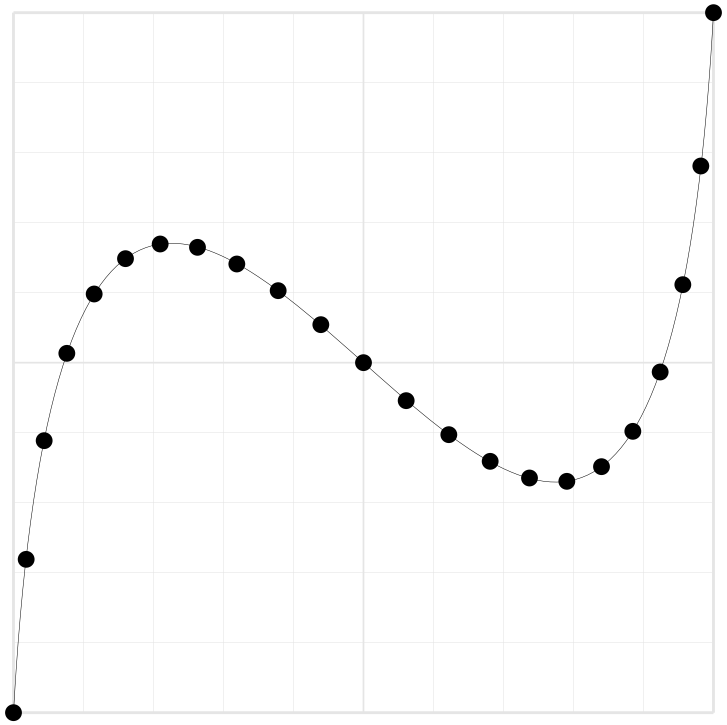
```
/quadpixel {transform 4 div round 4 mul itransform} def

/setgrid {save /rubbersnap exch def quadpixel /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{ 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

% pcurveto grabs the parameters for a curveto

/pcurveto {6 copy /y3 exch def /x3 exch def /y2 exch def /x2 exch def /y1 exch def
/x1 exch def currentpoint /y0 exch def /x0 exch def curveto} def

% xtt finds x(t) given t in the range of 0 to 1 ...

/xtt {x3 x2 3 mul sub x1 3 mul add x0 sub tt 3 exp mul x2 3 mul x1 6 mul neg
add x0 3 mul add tt dup mul mul add x1 3 mul x0 3 mul neg add tt mul add x0
add} def

% ytt finds y(t) given t in the range of 0 to 1 ...

/ytt {y3 y2 3 mul sub y1 3 mul add y0 sub tt 3 exp mul y2 3 mul y1 6 mul neg add
y0 3 mul add tt dup mul mul add y1 3 mul y0 3 mul neg add tt mul add y0 add}
def

% plotdots is used in the demo to show the position of t along the curve.
% In general, t changes more quickly along the "more bent" portions.

/numpoints 100 def % default - usually fast and 0.1 percent accurate
/dotsize 0.12 def  % default

/plotdots {0 1 numpoints 1 sub div 1.00001 {/tt exch def xtt ytt dotsize 0 360 arc
fill} for }def
```

For more help:

(520) 428-4073

Bezier curve length...

```
% bezierlength finds the length of a Bezier curve routed through pcurveto and
% leaves that length on the stack. It works by breaking the curve up into straight
% line approximate segments.

/bezierlength {/oldx x0 def /oldy y0 def /blength 0 def 0 1 numpoints 1 sub div
1.0001 {/tt exch def xtt ytt /newy exch def /newx exch def newx oldx sub dup mul
newy oldy sub dup mul add sqrt blength add /blength exch def /oldy newy def
/oldx newx def} for }def

%  //// demo - remove or alter before reuse ////

100 200 30 setgrid 10 10 showgrid     % Draw a pretty 20 x 20 grid

% Draw a desired single Bezier curve, saving key values ...

0 setlinewidth
0 0 moveto
1 18 9 -8 10 10 pcurveto    % your Bezier curve goes here
stroke

% Show dots for incremental value of Bezier z from 0 to 1 ...

/numpoints 23 def         % pick the number of segments
plotdots                  % and plot them

% Find the approximate length of the curve and report to host ...

1500 {37 sin pop} repeat  % optional host reporting delay

bezierlength              % calculate length
blength == flush          % report to host

% reported length for the curve shown:   21.585
```



Don Lancaster's PostScript Secrets

Bonus Supplement #35-B

Bezier length

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

36

Menu font lotsadots
That *Adobe Distillery*
Badge centering tricks
Repeat procs for forms
Scoring and perforating
Passing beyond *restore*

A PostScript save and restore does *not* affect the stack contents. To pass any variables or other information beyond a *restore*, just pile them up on the stack and then post-restore redefine them.

These two simple and powerful repeat commands can greatly ease building up all your PostScript forms and charts...

```
/xrpt {gsave aload pop /xtrips exch def /xdist exch def /xrproc exch def
xtrips {gsave xrproc grestore xdist 0 translate } repeat grestore} def

/yrpt {gsave aload pop /ytrips exch def /ydist exch def /yrproc exch def
ytrips {gsave yrproc grestore 0 ydist translate } repeat grestore} def

%  //// demo - remove before use. ////

[{200 300 moveto 160 0 rlineto stroke} 20 6] yrpt
[{200 300 moveto 0 100 rlineto stroke} 40 5] xrpt showpage
```

The proc in the zero array position gets spaced by the first array value and repeated by the second.

To improve the centering on your badges, place an extra 4-3/4 inch (345 point) outside circle around each printed badge image. This will fit the *outside* of the *Badge-a-Minit* circle cutter tool.

MATERIALS OF THE MONTH —

Sources of low cost **SCORING, SLITTING**, and **PERFORATING** ideas now include *American Safety Razor* at (703) 248-8000, *Sandco* at (918) 584-2985, *H.S. Boyd* (918) 835-9359, *Tandy* (817) 551-9770, *International Knife* (800) 345-9872, *Atlas Steel Rule Die* (800) 255-8786, and *Printer's Shopper* at (800) 854-2911.

PUBLICATION OF THE MONTH —

The **ADOBE DISTILLERY** will perform pseudo-compiling for you, and often can both shorten and speed up nearly all PostScript code. It is also a great demo and learning tool. From *Adobe Systems*, at 1585 Charleston Road, in Mountain View, CA, 94039. (415) 961-4400.

Free Font	**$0.00**
Not So Free Font	**$14.95**
Definitely Unfree Font	**$395.50**

The secret to a **MENU FONT** is to put all of the dots down first and then erase the ones you don't really want. Like so...

```
/menuline {gsave /rightstring exch def /leftstring exch def xpos ypos
translate 0 0 moveto 0 dotstring stringwidth pop dup 2 div txtwide exch
sub {pop dotstring show} for eraseheight setlinewidth 1 setlinecap 1 setgray
0 0 moveto leftstring stringwidth pop 0 rlineto stroke txtwide rightstring
stringwidth pop sub 0 moveto rightstring stringwidth pop 0 rlineto stroke 0
setgray 0 0 moveto leftstring show txtwide rightstring stringwidth pop sub
0 moveto rightstring show 0 setgray /ypos ypos yinc sub def grestore} def

%  //// demo - remove before use. ////

/NewCenturySchlbk-Bold findfont 10 scalefont setfont

/xpos 150 def              % left margin
/ypos 300 def              % base position of top line
/yinc 14 def               % vertical spacing between lines
/txtwide 250 def           % width of final line
/eraseheight 5 def         % width to guarantee dot erasure
/dotstring ( .) def        % dot pattern to lay down

(Free Font) ($0.00) menuline
(Not So Free Font) ($14.95) menuline
(Definitely Unfree Font) ($395.50) menuline  showpage quit
```

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #36-A

Menu justification

Mitzi's Menu...

A further demo of the gonzo justification routines that includes an automatic and precise menu justify and an astonishingly serendipitious 170 byte PostScript border.

(A) WIDTH CALCULATIONS

A new curwide variable lets you calculate the gonzo string width independent of printing it. Handy for menu justify, etc.. Use has to be bracketed with /oktoprint false def /oktoadvance false def --- stuff --- and then true. Works with any justify mode.

Make this change to main code so lengths can be extracted. Note that roomleft gets flushed at line end.

```
gonzo begin
/endtheline {/curwide txtwide roomleft sub def justx cvx exec oktoprint {printline}
if} bind def

% calloutwidth calculates only the width of the string, returning it to the top of
% the stack. For consistency with other callouts, use 0 0 (string) cw. This
% INCLUDES all stretching and font changes!

/cw {save /snapc1 exch def /oktoadvance false def /oktoadvance false def
/linestring linestring2 def /justx (justL) def 3 1 roll /ypos exch def /xpos exch def
stringgonzo curwide snapc1 restore} def
end
```

(B) MENU JUSTIFY

The cm routine takes a leading string and a trailing string separated by a delimiter and places a row of CONSTANT WIDTH, PRECISELY SPACED, VERTICALLY ALIGNED and WHOLE dots (or whatever) between the two. It is based on putting down the dots first and ERASING only whole dots that are not wanted.

This has been modified specially for a 10X grid

```
/menudots ( . ) def
/menufont {font1} def
/mdoteht 0.3 def
/menudelim (   ) def
/cropleadingspaces true def

% drawmdots draws a line of menu dots, using plain old show

/drawmdots { gsave menufont xxm yym moveto txtwide menudots stringwidth pop
dup /mdot1 exch def div floor cvi {menudots show} repeat grestore } def

% spchomp takes a string and conditionally removes all leading spaces, returning
% the truncated string

/spchomp {cropleadingspaces {dup 0 exch {32 eq {1 add}{exit} ifelse } forall} if exch
dup length 2 index sub 3 -1 roll exch getinterval} def

% mlineproc takes the mline string and processes it into a leading string, the dot
% row, and a trailing string. Unused dots are erased with logic to insure whole
% dots only. Null strings are ignored.

/mlineproc {mline length 0 gt {drawmdots mline menudelim search {/lmstr exch
def pop spchomp /rmstr exch def } if gsave 1 setgray xxm yym moveto mdoteht
setlinewidth 0 0 lmstr cw mdot1 div 10 mul ceiling 10 div % for 10X grid only
mdot1 mul 0 rlineto stroke xxm txtwide mdot1 div floor mdot1 mul add yym
moveto 0 0 rmstr cw mdot1 div 10 mul ceiling 10 div mdot1 mul % for 10X grid
only neg 0 rlineto stroke grestore xxm yym lmstr cl xxm txtwide add yym rmstr cr
/yym yym yinc sub def} if} def
```

For more help:
(520) 428-4073

Mitzi's menu...

```
% mj menujustify accepts tabbed string pairs and converts them into dotted
% menu listings..

/cm {gsave  /msg exch def  /yym exch def /xxm exch def { msg () search {/mline
exch def pop /msg exch def mlineproc} {/mline exch def mlineproc exit } ifelse}
loop grestore} def

%  //// demo - remove or alter before reuse. ///

gonzo begin                 % turn on the preloaded gonzo justification
gutil begin                 % turn on the preloaded gonzo utilities
printerror                  % show any errors to the page
nuisance begin              % activate the nuisance dictionary

/showthegrid false def       % view the layout grid?
20 20 10 setgrid               % get on the layout grid
showthegrid {50 50 showgrid} if   % and maybe show it

9 41 mt 5 pu 29 pr 5 pd closepath
bestgray lightgray fill black       % title fill goes UNDER border

% This is the "magic" border. For other effects, change the roundpath radius to
% 0.01, 0.8, 1, 1.5, 2.5, 3.5, 5 or 8. Or -2 or -6 (!)
% DO NOT USE 3.0. Use 2.99 instead.

1 setlinejoin 1 setlinecap
2.99 [20 12 6 12 6 9 9 9 9 49 6 49 6 46 41 46 41 49 38 49 38 9 41 9 41 12 20
12] roundpath closepath
[0.8 0 0.5 1  0.28 0 0.1 1] superstroke

% this draws the title bar and autoerases the ends

0 setlinejoin 0 setlinecap
0.01 [9 42 9 41 38 41 38 42 38 40 38 41 9 41 9 40] roundpath closepath
[0.8 0 0.5 1  0.28 0 0.1 1] superstroke

% set up the menu stuff ...

/txtwide 22 def                         % textwidth
/cstretch 0.01 def                      % extra character kerning
/sstretch 0.05 def                      % extra space kerning
/yinc 2.6 def                          % line spacing

/font1 /Palatino-Roman 1.2 gonzofont    % two nice yuppy fonts
/font2 /Palatino-Italic 1.2 gonzofont   % Use Bensquat if you have it
/font3 /ZapfChancery-MediumItalic 2.5 gonzofont  % for header

% Trick to cure Mitzi's anoxerism ...

font3
23.5 42.7 (Mitzi's   Yuppy   Fare) cc
23.5 42.75 (Mitzi's   Yuppy   Fare) cc
23.5 42.8 (Mitzi's   Yuppy   Fare) cc

font1
12.5 36.5
(Knockwurst Ala King                 $3.27
Hot Clam Sundae                      $1.15
Coconut Anchovy Pizza                  $8.35
Butterscotch Pudding au gratin      $1.25
Pastrami & Kiwi Blintz               $4.50
Sopapillas, Enchilada Style          $2.46
Grape Chili au jus              $4.45
Avocado & Liver Surprise                 .15
Chocolate Sushi                     $25.50
) cm

showpage quit
```

Don Lancaster's PostScript Secrets

Bonus Supplement #36-B

Menu justification

For more help:
(520) 428-4073

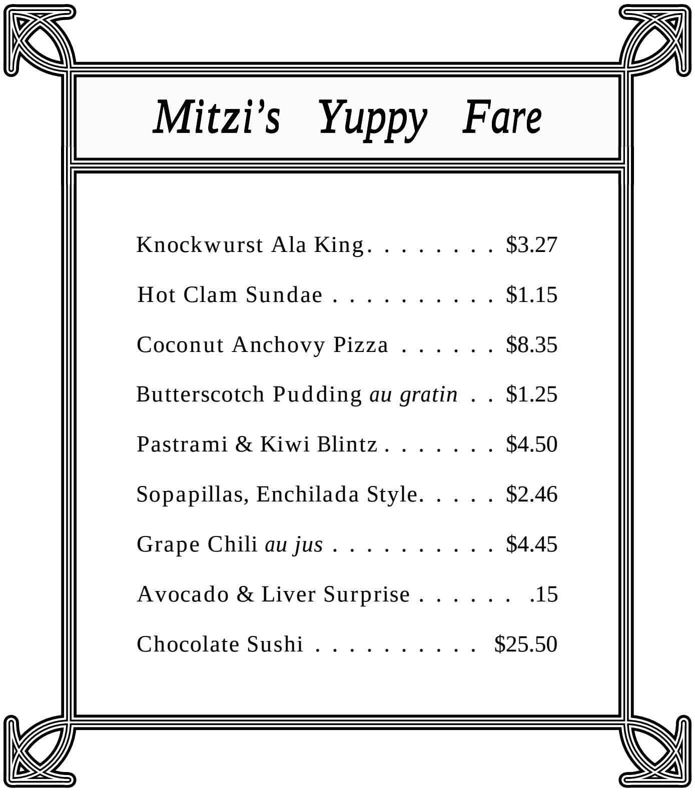
Don Lancaster's PostScript Secrets

Bonus Supplement #36-C

Menu justification

Mitzi's Menu...

This is what our menu actually looks like



Mitzi's Yuppy Fare

Knockwurst Ala King	$3.27
Hot Clam Sundae	$1.15
Coconut Anchovy Pizza	$8.35
Butterscotch Pudding *au gratin*	$1.25
Pastrami & Kiwi Blintz	$4.50
Sopapillas, Enchilada Style	$2.46
Grape Chili *au jus*	$4.45
Avocado & Liver Surprise	.15
Chocolate Sushi	$25.50

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

37

Here's a simple center justify routine...

```
/simplecj {dup stringwidth pop 2 div
4 -1 roll sub neg 3 -1 roll moveto show} def

/Helvetica findfont [24 0 0 24 0 0] makefont setfont
306 500 (CENTERED HEADLINE) simplecj showpage quit
```

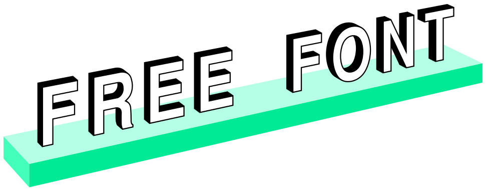
The **306** is the centering line and the **500** is the vertical position. For a simple right justify, eliminate the **2 div** from the above.

MATERIAL OF THE MONTH —

New **ETCH'N PEEL** photosensitized polyester masking films now use PostScript instead of a knife. Just expose, pour on glop, wash, and you are cut to size. For traditional graphics arts uses, for back-lit sign lettering, bunches more. From *Kimoto USA*, 2915 182nd Street, Redondo Beach, CA 90278. (213) 370-7411.

PUBLICATION OF THE MONTH —

That new **BLACK BOOK**, or the **ADOBE TYPE I FONT FORMAT** tells all. From *Adobe Systems*, 1585 Charleston Road, Mountain View, CA, 94039. (415) 961-4400.



Now that font paths are grabbable, genuine 3-D perspective is a snap...

```
/T {611 {14 729 mt 14 604 li 235 604 li 235 0 li 385 0 li 385 604 li 598 604 li
598 729 li cp}} def /N {722 {68 729 mt 68 0 li 218 0 li 218 508 li 513 0 li 661 0
li 661 729 l 511 729 li 511 228 li 216 729 li cp}} def /O {778 {742 359 mt 742
469 702 579 634 650 ct 573 713 484 741 391 741 ct 298 741 209 713 148
650 ct 80 579 40 469 40 359 ct 40 249 80 139 148 68 ct 209 5 298 -23 391
-23 ct 484 -23 573 5 634 68 ct 702 139 742 249 742 359 ct cp 391 108 mt
343 108 295 122 257 157 ct 209 201 185 280 184 359 ct 185 438 209 517
257 561 ct 295 596 343 610 391 610 ct 439 610 487 596 525 561 ct 573 517
597 438 598 359 ct 597 280 573 201 525 157 ct 487 122 439 108 391 108 ct
cp}} def /F {611 {74 729 mt 74 0 li 224 0 li 224 313 li 543 313 li 543 438 li
224 438 li 224 604 li 586 604 li 586 729 li cp}} def /E {667 {79 729 mt 79 0 li
624 0 li 624 125 li 229 125 li 229 313 li 578 313 li 578 438 li 229 438 li 229
604 li 607 604 li 607 729 li cp}} def /R {722 {80 0 mt 230 0 li 230 288 li 406
288 li 458 288 494 264 496 215 ct 494 162 494 104 496 68 ct 498 44 504
22 516 0 ct 677 0 li 677 27 li 661 37 649 46 649 87 ct 648 124 646 220 642
250 ct 634 316 586 334 561 351 ct 610 380 642 403 660 466 ct 678 529 663
581 653 613 ct 632 692 558 726 485 729 ct 80 729 li cp 230 604 mt 433 604
li 500 604 524 574 524 522 ct 533 450 470 413 435 413 ct 230 413 li cp}}
def /space {250 {}} def

/mt {exch fontsize 0 get mul .001 mul xoffset add exch fontsize 3 get mul .001
mul yoffset add nlt moveto} def /li {exch fontsize 0 get mul .001 mul xoffset
add exch fontsize 3 get mul .001 mul yoffset add nlt lineto} def /ct {3 {6 2 roll
exch fontsize 0 get mul .001 mul xoffset add exch fontsize 3 get mul .001 mul
yoffset add nlt} repeat curveto} def /cp {closepath} def /strrx ( ) def /nlt
{currenty exch prd1} def /charproc {gsave 1 setgray fill grestore stroke} def

/nlshow { /msg exch def msg {strrx exch 0 exch put strrx dup ( ) eq {pop
(space) } if cvn load exec exec charproc fontsize 0 get mul .001 mul extrakern
add xoffset add /xoffset exch def} forall} def /prd1 {/zi exch def /yi exch def /xi
exch def objrot cos xi mul objrot sin yi mul sub objrot sin xi mul objrot cos yi
mul add zi /zz exch def /yy exch def /xx exch def yo dup yy add div dup xx xo
sub mul exch zz zo sub mul} def

%  //// demo - remove before use. ////

/fontsize [72 0 0 72 0 0] def /extrakern 10 def /xo -300 def /yo 700 def /zo
500 def /objrot 15 def -150 900 translate 6 -.6 0 {/currenty exch def /xoffset 0
def /yoffset 0 def (FREE  FONT) nlshow } for showpage quit
```

A font path grabbing routine appears on the next page.

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #37A

Level 2
font path grabber

PostScript level 2 font path grabber...

There are lots of exciting and unique new features to PostScript Level 2 that makes this a "must have" upgrade. But by far the most useful (and long overdue) feature of Level 2 is that nearly all font paths are fully open and unlocked. And thus easily captured.

Font paths are simply and quickly grabbable on the fly. Besides the obvious use of creating outline characters, you can use captured font paths for composite characters, to eliminate the need for font downloads on a logo or letterhead, to do the non-linear transformations needed for perspective, 3-D, star wars and other distorted lettering, and to capture letters for vinyl sign cutters and wood routers.

Because of the **invalidaccess** lockout on earlier versions of Post Script, earlier font path grabbing was complex and tricky. With Level 2, you simply grab the path on the fly.

I will show you only the bare fundamentals of font path grabbing here; see our many other PSRT library files for detailed examples of the incredible stuff you can do with grabbed paths.

THE ROUTINES SHOWN HERE WORK ONLY ON POSTSCRIPT LEVEL TWO. But you can easily take the resultant paths and use them on any genuine Adobe PostScript printer of any level.

First, define your font. If you are going to work with the usual font dictionaries, you may want to make it 1000 points high. Otherwise, pick your final size...

```
/NewCenturySchlbk-Bold findfont [160 0 0 130 0 0] makefont setfont
```

Then select a letter and just grab your fontpath...

```
0 0 moveto (s) false charpath
```

This now creates an unlocked outline path. If you want to, you can combine the path with other design elements.

Here is how to convert the path to an array. An array is chosen because it is easier to report back to your host.

```
mark

{/moveto cvx}
{/lineto cvx}
{/curveto cvx}
{/closepath cvx}

pathforall

]   /outlinepath exch def
```

And it is that simple! If you want to do non-linear transformations or otherwise leap tall buildings in a single bound, you just alter the four procs needed by pathforall.

Here is how you report your path back to your host for permanent recording. Naturally, you ARE using full two-way host recordable comm or you absolutely, positively, emphatically should not be here.

```
/stall {{37 sin pop} repeat} def         % useful delay proc

2500 stall                               % optional host turnaround delay

outlinepath {== flush 50 stall } forall  % optional internal character delay
```

For more help:
(602) 428-4073

Here is your expected host returned data for our example...

```
19.52 2.35001 moveto
25.12 -0.509979 35.52 -2.06998 42.08 -2.06998 curveto
45.44 -2.06998 50.56 -1.41998 53.76 -0.769989 curveto
60.48 0.790009 63.68 1.95999 68.48 6.12 curveto
72.8 9.89001 74.56 15.09 74.56 19.9 curveto
74.56 26.53 70.4 32.25 63.36 35.37 curveto
58.08 37.71 51.52 38.88 45.6 40.05 curveto
38.08 41.61 lineto
33.44 42.52 24.48 43.3 24.48 48.5 curveto
24.48 54.61 32.16 56.43 38.56 56.43 curveto
49.6 56.43 57.44 49.93 62.72 42.78 curveto
63.36 42.39 63.36 41.74 63.84 41.35 curveto
70.88 41.35 lineto
70.24 61.63 lineto
63.84 61.63 lineto
57.76 58.64 lineto
52.32 61.24 44.8 62.93 38.4 62.93 curveto
22.72 62.93 7.19998 56.3 7.19998 42.26 curveto
7.19998 36.54 9.12 32.38 14.24 28.61 curveto
20.0 24.32 24.96 23.28 32.32 21.98 curveto
42.4 20.29 lineto
50.24 18.86 59.2 16.91 55.04 8.46002 curveto
52.0 5.21002 46.72 4.29999 41.92 4.29999 curveto
27.68 4.29999 18.88 12.62 13.12 22.11 curveto
6.07999 22.11 lineto
6.87997 -0.119995 lineto
13.12 -0.119995 lineto
closepath

80.0 0.0 moveto
```

Note that the last moveto is really your x and y character spacing to the next character.

You can obviously edit the host returned and recorded data to a form of

/s {lots of movetos, linetos, etc...} def

Note that this form is MUCH shorter than towing an entire font around if you just need a few wierd letters for a logo or whatever.

Don Lancaster's PostScript Secrets

Bonus Supplement # 37B

Level 2
font path grabber

For more help:
(602) 428-4073

Bonus Supplement #37C

Level 2
font path grabber

```
% SOME POSSIBLE LEVEL 1 PROBLEMS:
% ==============================

% Yes, you can use full messages instead of individual letters on
% Level 2. But for Level 1 compatibility, LIMIT YOUR FONT GRABS TO
% INDIVIDUAL SINGLE CHARACTERS.

% PostScript Level 2 dynamically allocates its memory, so there is no
% sane limit to the amount of stuff you can put on your stack at one time.

% But PostScript Level 1 only allows 512 characters on the stack at once.
% If your font path exceeds 450 or so characters, you may get a stack
% overflow on Level 1.

% Besides an obvious counting, here is another way you can check the length
% of your captured font path...

          outlinepath length == flush

% In this example, a "153" should have been returned to your host screen
% (You ARE using two-way comm, of course). Which tells us that this
% path is "level 1 safe".

% If your captured font path is too long, change it to this form...

%   /mycapturepath {
%
%          { 400 or fewer stack items} cvx
%          { 400 or fewer stack items} cvx
%          { 400 or fewer stack items} cvx
%            ....           ...
%          { 400 or fewer stack items} cvx
%
%                } def


% This should run just fine on any Level 1 or Level 2 machine.

% We will be doing LOTS more with path grabs, so be sure to follow PSRT.
```

To minimize extra heavy paper jams on an SX printer, use only your "straight through" paper path, keep your guide rails loose, and any sheet curl down. Then, before you actually print, manually force your sheet 1/8 of an inch onto the pickup roller and uniformly pull it back half of that. Card stock and heavy parchment should now feed ok.

Here's a useful character border routine...

```
❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑
/stringup {/char exch def {gsave char show grestore
currentpoint vcharsp add moveto} repeat} def

/stringright {/char exch def {gsave char show grestore
currentpoint exch hcharsp add exch moveto} repeat} def

/stringdown {/char exch def {gsave char show grestore
currentpoint vcharsp sub moveto} repeat} def

/stringleft {/char exch def {gsave char show grestore
currentpoint exch hcharsp sub exch moveto} repeat} def

% /// demo - remove before use. ///

/ZapfDingbats findfont [12 0 0 12 0 0] makefont setfont
20 30 moveto /vcharsp 20 def /hcharsp 20 def
24 (q) stringup 13 (q) stringright 24 (q) stringdown
13 (q) stringleft showpage quit
❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑❑
```

A little known use for the *Kroy Color* process is to instantly create a totally opaque negative. Simply Kroy Color in the usual way. Then throw out the baby and keep the washwater. Do watch carefully for fills on ultra-fine detail.

MATERIAL OF THE MONTH —

The new **PELSAER HOT BINDING** process consists of a glue preform and two fly sheets. The cost is around 50 cents each. Yes, you can now perfect bind *any* cover material and even print on the spine. From *Leonard's Unibind/Pelsaer* 4125 Prospect Drive, Carmichael, CA 95608. (916) 967-6401.

PUBLICATIONS OF THE MONTH —

A handy directory of **1500 POSTSCRIPT TYPEFACES** is available for $9.95 through *The Font Company* at (800) 442-FONT. Other type directories are available from *Adobe* at (415) 961-4400; from *Gerber* found at (800) 222-7446; *International Typeface* at (212) 371-0699; *Kingsley ATF* at (602) 325-5884; or *Monotype* at (312) 855-1440.

• • • FREE FONT • • •

Most PostScript fonts have hidden characters which are not normally accessable by a single keystroke. Here's some bullets...

```
/AvantGarde-Demi findfont [30 0 0 30 0 0] makefont setfont
100 200 moveto (\267 \267 FREE  FONT \267 \267) show showpage quit
```

And here are several other "shy" characters...

\261 on most fonts is an em dash –
\267 on most fonts is a bullet •
\274 on most fonts is the ellipsis ...
\320 on most fonts is an en dash —

\251 on the Symbol font is a heart ♥
\323 on the Symbol font is a copyright ©
\324 on the Symbol font is a trademark ™
\360 on the Symbol font is the apple ONLY on LaserWriters.

Use the Red Book or a full 256 character dump to find the others. Note that you must provide a reverse slash before any *unbalanced* **)** or **(**.

Don Lancaster's PostScript Secrets

38

Heavy paper jams
Kroy Color negatives
Quick border routine
Hidden font characters
1500 typeface directory
Pelsaer thermal binding

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #38

A nonlinear *ashow* operator

A non-linear ashow operator...

This routine is similar to the PostScript *ashow* operator, except that you can non-linearly remap your text message onto any surface. Use this for perspective, star wars, pennants, spheres, etc...

```
% Here are some typical font paths...

/T {611 {14 729 mt 14 604 li 235 604 li 235 0 li 385 0 li 385 604 li 598 604
li 598 729 li cp}} def

/N {722 {68 729 mt 68 0 li 218 0 li 218 508 li 513 0 li 661 0 li 661 729 li
511 729 li 511 228 li 216 729 li cp}} def

/O {778 {742 359 mt 742 469 702 579 634 650 ct 573 713 484 741 391
741 ct 298 741 209 713 148 650 ct 80 579 40 469 40 359 ct 40 249 80
139 148 68 ct 209 5 298 -23 391 -23 ct 484 -23 573 5 634 68 ct 702 139
742 249 742 359 ct cp 391 108 mt 343 108 295 122 257 157 ct 209 201
185 280 184 359 ct 185 438 209 517 257 561 ct 295 596 343 610 391 610
ct 439 610 487 596 525 561 ct 573 517 597 438 598 359 ct 597 280 573
201 525 157 ct 487 122 439 108 391 108 ct cp}} def

/F {611 {74 729 mt 74 0 li 224 0 li 224 313 li 543 313 li 543 438 li 224 438
li 224 604 li 586 604 li 586 729 li cp}} def

/E {667 {79 729 mt 79 0 li 624 0 li 624 125 li 229 125 li 229 313 li 578 313
li 578 438 li 229 438 li 229 604 li 607 604 li 607 729 li cp}} def

/R {722 {80 0 mt 230 0 li 230 288 li 406 288 li 458 288 494 264 496 215 ct
494 162 494 104 496 68 ct 498 44 504 22 516 0 ct 677 0 li 677 27 li 661
37 649 46 649 87 ct 648 124 646 220 642 250 ct 634 316 586 334 561
351 ct 610 380 642 403 660 466 ct 678 529 663 581 653 613 ct 632 692
558 726 485 729 ct 80 729 li cp 230 604 mt 433 604 li 500 604 524 574
524 522 ct 533 450 470 413 435 413 ct 230 413 li cp}} def

/space {250 {}} def

% Your non-linear transform goes here. It accepts x and y on the stack and
% later will return x' and y' back to the stack, following your nonlinear
% remapping rule or rules. The transform is used once for a lineto and
% three times for a curveto...

/nlt {} def    % substitute your own nonlinear operator here

% Service routines...

/mt {exch fontsize 0 get mul .001 mul xoffset add nlt exch fontsize 3 get
mul .001 mul yoffset add moveto} def  % nonlinear moveto

/li {exch fontsize 0 get mul .001 mul xoffset add nlt exch fontsize 3 get mul
.001 mul yoffset add lineto} def  % nonlinear lineto

/ct {3 {6 2 roll exch fontsize 0 get mul .001 mul xoffset add nlt exch fontsize
3 get mul .001 mul yoffset add} repeat curveto} def % nonlinear curveto

/cp {closepath} def

/strrx ( ) def

/nlashow {/msg exch def msg {strrx exch 0 exch put strrx dup ( ) eq {pop
(space)} if cvn load /curchar exch def curchar 0 get /cwide exch def curchar
1 get /nlashow {/msg exch def msg {strrx exch 0 exch put strrx dup ( ) eq
{pop (space) } if cvx exec exec charproc fontsize 0 get mul 1000 div
extrakern add xoffset add /xoffset exch def} forall} def

/charproc {stroke} def  % does what you want to your path

% //// demo - remove before use. ////

/fontsize [72 0 0 72 0 0] def   % font matrix
/xoffset 100 def                % starting horizontal position
/yoffset 200 def                % starting vertical position
/extrakern 3 def                % additional spacing between characters

(FREE  FONT) nlashow

showpage quit
```

Note that this method is much faster than pixel line remapping, but you do have to previously capture the plaintext font path of every character you want to use. We'll be seeing several new examples of nonlinear transformations in future supplements. For now, why not dream up some on your own?

For more help:

(520) 428-4073

Don Lancaster's PostScript Secrets

39

College pennant font

Pad printing resources

Standard Rate and Data

Targeted Marketing lists

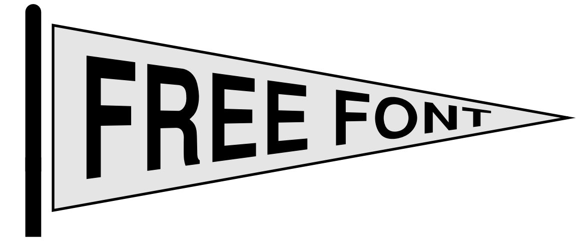
Nonlinear transform use

MATERIALS OF THE MONTH —

Some sources of those **PAD PRINTERS** that can print on just about anything include *Barton, Nelson* (800) 821-6697, *Basco* (818) 718-1506; *Enercon* (414) 255-6070; *Print Central* (314) 364-0339; *Service Tectonics* (417) 263-0758; and *Wagner* (314) 257-3908. A good summary of pad printing appeared in the April 1989 issue of *Screen Printing* magazine (513) 421-2050.

PUBLICATIONS OF THE MONTH —

The best sources for **MAILING LISTS** to buy, sell, rent, or trade are available through *Standard Rate and Data Service* at 3004 Glenview Road, Wilmette, IL 60091. (800) 323-4588. Since all of those SR&D directories are expensive, you may want to try a library or pick up out-of-date copies through your local ad agency. One leading source of computer related mailing lists is Irv Brechner's *Targeted Marketing*, Box 453, Livingston, NJ 07029. (201) 731-4382.



Feeling pennitant this month? Yea team...

```
/T {611 {14 729 m 14 604 l 235 604 l 235 0 l 385 0 l 385 604 l 598 604 l 598
729 l p}} def /N {722 {68 729 m 68 0 l 218 0 l 218 508 l 513 0 l 661 0 l 661
729 l 511 729 l 511 228 l 216 729 l p}} def /O {778 {742 359 m 742 469 702
579 634 650 c 573 713 484 741 391 741 c 298 741 209 713 148 650 c 80
579 40 469 40 359 c 40 249 80 139 148 68 c 209 5 298 -23 391 -23 c 484
-23 573 5 634 68 c 702 139 742 249 742 359 c p 391 108 m 343 108 295
122 257 157 c 209 201 185 280 184 359 c 185 438 209 517 257 561 c 295
596 343 610 391 610 c 439 610 487 596 525 561 c 573 517 597 438 598
359 c 597 280 573 201 525 157 c 487 122 439 108 391 108 c p}} def /F {611
{74 729 m 74 0 l 224 0 l 224 313 l 543 313 l 543 438 l 224 438 l 224 604 l
586 604 l 586 729 l p}} def /E {667 {79 729 m 79 0 l 624 0 l 624 125 l 229
125 l 229 313 l 578 313 l 578 438 l 229 438 l 229 604 l 607 604 l 607 729 l
p}} def /R {722 {80 0 m 230 0 l 230 288 l 406 288 l 458 288 494 264 496 215
c 494 162 494 104 496 68 c 498 44 504 22 516 0 c 677 0 l 677 27 l 661 37
649 46 649 87 c 648 124 646 220 642 250 c 634 316 586 334 561 351 c 610
380 642 403 660 466 c 678 529 663 581 653 613 c 632 692 558 726 485
729 c 80 729 l p 230 604 m 433 604 l 500 604 524 574 524 522 c 533 450
470 413 435 413 c 230 413 l p}} def /space {250 {}} def

/fudge1 {1 xoffset fudge mul sub mul} def

/m {exch fontsize 0 get mul .001 mul fudge1 xoffset add exch fontsize 3 get
mul .001 mul yoffset add nlt moveto} def /l {exch fontsize 0 get mul .001 mul
fudge1 xoffset add exch fontsize 3 get mul .001 mul yoffset add nlt lineto} def
/c {3 {6 2 roll exch fontsize 0 get mul .001 mul fudge1 xoffset add exch
fontsize 3 get mul .001 mul yoffset add nlt} repeat curveto} def /p {closepath}
def /strrx ( ) def  /charproc {fill} def

/nlshow { /msg exch def msg {strrx exch 0 exch put strrx dup ( ) eq {pop
(space) } if cvn load exec exec charproc fontsize 0 get mul 0.001 mul
extrakern add fudge1 xoffset add /xoffset exch def} forall} def

%  //// demo - remove or alter before reuse. ////

/fontsize [72 0 0 130 0 0] def /extrakern 2 def /fudge 1 600 div def /nlt {/yyin
exch def /xxin exch def xxin 350 exch sub 350 div yyin mul xxin exch} def 100
200 translate 106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen

-20 0 moveto 0 70 rlineto 390 -70 rlineto -390 -70 rlineto closepath gsave 0.9
setgray fill grestore 2 setlinewidth stroke

12 setlinewidth -35 -90 moveto 0 60 rlineto 0 setlinecap stroke -35 -40
moveto 0 120 rlineto 1 setlinecap stroke

/xoffset 0 def /yoffset -47 def 1 setlinewidth (FREE FONT) nlshow showpage
quit
```

For more help:

(520) 428-4073

Don Lancaster's
PostScript
Secrets

Bonus
Supplement
#39

More Details on
Pad Printing

More Details on Pad Printing...

Pad printing is an arcane and very sneaky process that allows you to full-color print onto such specialty objects as pens, golf balls, electronic keycaps, classic car speedometers, antique radio dials, or even on eggshells.

Obviously, PostScript cries to be combined with a cheap do-it-yourself home pad printing operation, to open up whole new worlds of ultra low cost, lower end specialty advertising products, sold direct and real time to the end user in sanely small quantities.

Here, briefly, is how pad printing works: An engraved plate known as a *cliche* contains a flat and frontwards image of your artwork. Ink is spread onto the plate and then doctored off, leaving an ink image. The ink image is picked up by a special shaped silicon rubber pad and then transferred to the irregular object being printed.

This transfer occurs because ink exposed to air creates a thin and tacky film. Initially, the tack will stick to the pad. During the pad motion, a second and stronger tack builds up on the exposed surface, which lets the ink stick to the final object better than it does the pad.

A "nesting" carrier holds the object you are printing. The shape of the pad is typically an involute of your final printing path, so that you usually obtain a distortion-free result. Dozens of pad shapes and sizes are common. Several colors can even be done at once.

Here are some pad printer manufacturers...

BARTON, NELSON
3201 Gillham Plaza
Kansas City, MO 64109
(800) 821-6697

BASCO
9351 DeSoto Avenue
Chatsworth, CA 91311
(818) 718-1506

ENERCON
W140 N9572 Fountain Blvd.
P.O. Box 773
Menomonee Falls, WI 53051
(414) 255-6070

PRINT CENTRAL
Box JJ
Rolla, MO 65401
(314) 364-0339

SERVICE TECTONICS
2827 Treat Street
Adrian, MI 49221
(517) 263-0758

WAGNER
#5 Capper Drive
Pacific, MO 63069
(314) 257-3908

TAMPOFLEX (Germany)
BoschstraBe 5
D-7257 Ditzingen
Postfach 31 17 40
D-7000 Stuttgart 31
(0 71 56) 80 14

And here's a few recent useful papers on pad printing...

Pad printing: Jack-of-all-trades in industrial decorating
Steve Duccilli, *Screen Printing*, April 89, pp 63-71.

Pad geometry
John Legat, *Screen Printing*, October 84, pp 92-96.

Pad printing equipment review
Editors, *Screen Printing*, October 84, pp 103-106.

An evaluation of a unique image transfer process
Tamas Grecska, *Screen Printing*, October 84, pp 157-165.

Pad-transfer printing: Soft touch in a tough market
David Karlyn, *Screen Printing*, August 82, pp 100-104.

Looking at pad transfer printing
Paul Wasserman, *Screen Printing*, October 79, pp 122-125.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

40

A keycap free font
Revised *Red Book II*
Useful sticker routine
NTX ROM upgrade 3.0
Hard coated SX drums

Apple has issued an interim version 3.0 ROM upgrade for the NTX. This prints text up to 20% faster, fixes the worst of the SCSI bugs, and does improve the emulations, but is *not* PostScript II. Around $120 from your Apple dealer. Note that version 3.0 *requires* Mac Font Utility version 2.0.

Here's a useful sticker routine...

```
/makeasticker {360 exch div /ang exch def 2 div /trough exch def 2 div
/crest exch def newpath 0 crest moveto 0 ang 360 ang sub 0.001 add {/posn
exch def posn ang 2 div add dup sin trough mul exch cos trough mul lineto
posn ang add dup sin crest mul exch cos crest mul lineto} for} def

% //// demo - remove or alter before final use. ////

gsave 300 100 translate 100 50 10 makeasticker stroke grestore showpage
quit
```

See *Supplement 40S* for several unique variations on this sticker theme.

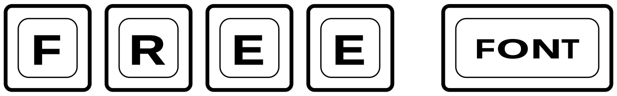
Note that the simplest way to pick up 600 DPI laser printing is to work double size and then photoreduce your jiffy-printed copies.

MATERIAL OF THE MONTH —

New **HARD COATED SX REPLACEMENT DRUMS** are becoming readily available and are good for as many as a dozen reloads without any problems. The leading source of these imported drums is *CopyMate Products*, at 20F Robert Pitt Drive, Monsey, New York, 10952. (800) 457-0074. Two sources of new hard drums that are preinstalled in refilled cartridges now include *Thompson and Thompson* at (714) 855-3838 and *Lazer Products* found at (303) 792-5277.

PUBLICATION OF THE MONTH —

Adobe's brand new **RED BOOK II**, or the **POSTSCRIPT REFERENCE MANUAL** now provides full details on PostScript Level II. Included are such goodies as filters, improved forms, FAX support, bunches more. *Addison Wesley* Publishing stock #18127. One stocking source is *Synergetics* at (602) 428-4073.



This keycap font is handy for user manuals and instruction sheets...

```
/n {newpath} def /m {moveto} def /c {curveto} def /y {0 exch rlineto} def /x {0
rlineto} def /s {stroke} def /w {setlinewidth} def /g {setgray} def

/keycap1 {gsave /msg exch def currentpoint 2 copy translate n 45 w 0 g 0
500 m 425 y 0 966 34 1000 75 1000 c 850 x 966 1000 1000 966 1000 925
c -850 y 1000 34 966 0 925 0 c -850 x 34 0 0 34 0 75 c 425 y s n 15 w 0 g
150 500 m 225 y 150 794 206 850 275 850 c 450 x 794 850 850 794 850
725 c -450 y 850 206 794 150 725 150 c -450 x 206 150 150 206 150 275
c 225 y s /AvantGarde-Demi findfont [750 0 0 500 0 0 ] makefont setfont
480 msg stringwidth pop 2 div sub 300 moveto msg show grestore exch
1200 add exch moveto } def

/keycap2 {gsave /msg exch def currentpoint 2 copy translate 45 w 0 g 5 500
m 425 y 4.6 966 38 1000 80 1000 c 1850 x 1971 1000 2005 966 2005 925
c -850 y 2005 34 1971 0 1930 0 c -1850 x 38 0 5 34 5 75 c 425 y s 0 g 15
w 155 500 m 225 y 155 794 211 850 280 850 c 1450 x 1799 850 1855 794
1855 725 c -450 y 1855 206 1799 150 1730 150 c -1450 x 211 150 155
206 155 275 c 225 y s /AvantGarde-Demi findfont [500 0 0 325 0 0 ]
makefont setfont 980 msg stringwidth pop 2 div sub 375 moveto msg show
grestore exch 1200 add exch moveto } def

% //// demo - remove or alter before use ////

100 200 moveto 45 1000 div dup scale (F) keycap1 (R) keycap1 (E) keycap1
(E) keycap1 currentpoint exch 400 add exch moveto (FONT) keycap2
showpage quit
```

For more help:
(602) 428-4073

Don Lancaster's LaserWriter Secrets

Bonus Supplement #40

Variations on the *Make-a-Sticker* theme

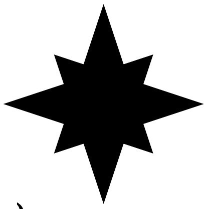
Variations on the Make-a-Sticker theme...

While originally written to create plain old stickers, the *Make-a-Sticker* code of *LaserWriter Secrets #40* can generate an incredible variety of output options.

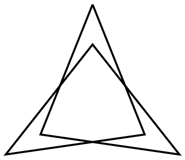
You do this simply by changing the crest, trough, and point values and then suitably stroking and filling. Very unusual patterns can result by using *negative* trough values. You can also try very large or very small point count values.

Quiz time. See if you can match the following demos with the stickers they generate...

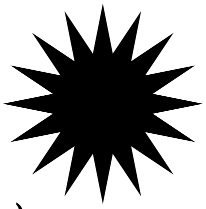
( ) 100 -70 4 makeasticker fill
( ) 100 -100 8 makeasticker fill
( ) 100 -60 3 makeasticker stroke
( ) 100 90 40 makeasticker stroke
( ) 100 50 10 makeasticker stroke
( ) 100 -50 10 makeasticker stroke
( ) 100 95 100 makeasticker stroke
( ) 100 -125 33 makeasticker stroke
( ) 100 -120 10 makeasticker stroke



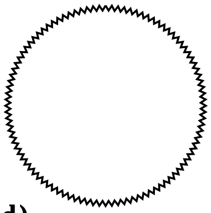
(a)



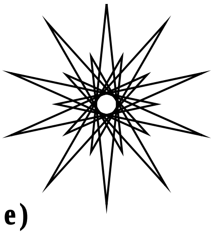
(b)



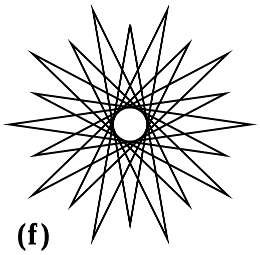
(c)



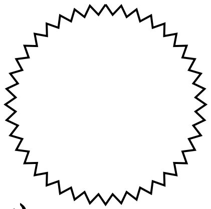
(d)



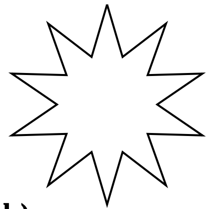
(e)



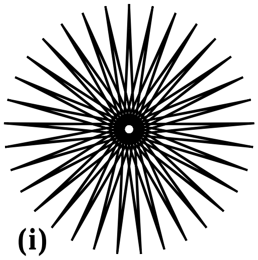
(f)



(g)



(h)



(i)

Which one of these is you?

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

41

VM errors
Photolabels
Hidden mirrors
New Gray Book
Starwars Free Font

There is a front surface mirror *above* the cartridge on SX printers such as the NTX. Any toner or dirt on the mirror can cause all sorts of subtle print quality problems. Carefully clean using a Q-Tip.

PostScript's *makefont* and *scalefont* operators are slow and burn up VM, but *setfont* is fast and clean. For maximum speed, predefine your "made" or "scaled" fonts, but do *not* use *setfont* until later.

The first thing to try on a *VM error* is a reboot to clear the machine of earlier downloads. Other cures do include writing clean code to start with, making use of *save/restore* reclamation, or adding extra RAM.

MATERIAL OF THE MONTH —

Peel and stick full color prints on *thin* paper by **PHOTOLABELS**. Prices from 12 cents each. Low cost color for calendars, car or house sale sheets, displays. *Photolabels*, 333 Kimberly Drive, Carol Stream IL 60188. (312) 690-0132.

PUBLICATION OF THE MONTH —

Glenn Reid's new **GRAY BOOK**, **THINKING IN POSTSCRIPT** gives solid treatment for PostScript-as-language and lots of programming constructs. *Addison Wesley* Publishing stock #52372. One stocking source is *Synergetics* (520) 428-4073.

FREE FONT
FREE FONT
FREE FONT

Here's a new, fast, and extremely accurate Starwars routine...

```
/mychardict 10 dict def mychardict dup

/F {480 {68 0 mt 205 0 li 205 303 li 433 303 li 433 436 li 205 436 li 205 606 li
444 606 li 444 739 li 68 739 li cp}} put dup /R {580 { 68 0 mt 205 0 li 205 283
li 399 0 li 568 0 li 368 273 li 418 284 463 305 499 344 ct 535 384 560 451 560
505 ct 560 574 527 643 474 687 ct 418 734 352 738 282 739 ct 68 739 li cp
205 387 mt 205 606 li 281 606 li 319 607 351 605 383 581 ct 409 562 423 530
423 498 ct 423 467 411 439 389 418 ct 362 392 316 387 280 387 ct cp}} put
dup /E {520 {68 0 mt 465 0 li 465 133 li 205 133 li 205 303 li 454 303 li 454
436 li 205 436 li 205 606 li 465 606 li 465 739 li 68 739 li cp}} put dup /space
{280 {280 0 mt}} put dup /O { 840 {415 753 mt 202 753 30 581 30 366 ct 30 152
205 -18 415 -18 ct 625 -18 800 152 800 366 ct 800 581 628 753 415 753 ct cp 41
5 620 mt 553 620 663 508 663 369 ct 663 232 550 115 415 115 ct 280 115 167
232 167 369 ct 167 508 277 620 415 620 ct cp}} put dup /N {740 {68 0 mt 205 0
li 205 538 li 515 0 li 667 0 li 667 739 li 530 739 li 530 199 li 222 739 li 68 739
li cp}} put dup /T {420 {143 0 mt 280 0 li 280 606 li 418 606 li 418 739 li 7 739
li 7 606 li 143 606 li cp}} put pop

/mt {exch fontsize 0 get mul .001 mul xoffset add exch fontsize 3 get mul .001
mul yoffset add nlt moveto} def /li {exch fontsize 0 get mul .001 mul xoffset add
exch fontsize 3 get mul .001 mul yoffset add nlt lineto} def /ct {3 {6 2 roll exch
fontsize 0 get mul .001 mul xoffset add exch fontsize 3 get mul .001 mul yoffset
add nlt} repeat curveto} def /cp {closepath} def

/swshow {/msg exch def /yoffset exch def /xoffset exch def /strrx (X) def msg
{strrx exch 0 exch put strrx dup ( ) eq {pop (space)} if cvn mychardict exch get
exec exec fill fontsize 0 get mul 0.001 mul extrakern add xoffset add /xoffset
exch def} forall} def /nlt {tiltfactor dup 2 index add div dup 4 -1 roll mul 3 1 roll
mul} def

%  //// demo - remove or alter before reuse. ////

200 200 translate /tiltfactor 224 def /fontsize [38 0 0 38 0 0 ] def /extrakern 1
def 0 0 moveto -95 15 (FREE FONT) swshow -95 57 (FREE FONT) swshow -95 99
(FREE FONT) swshow showpage quit
```

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #41-A

Gonzo
keystone justify

Gonzo Keystone Justify...

Some alpha release working additions to GONZO13A.PS that do extremely high quality left, center, or right keystone justification for blurbs, ad copy, or point-of-purchase signs. The routines are extremely fast and work in a single (or at most two) passes.

Required: Persistent download of GONZO13.PTL #219.

This .GPS (Guru PostScript) file DEMANDS the persistant download of GONZO13A.PS #219. GUTILITY.PS #102 is also recommended.

This is a PRELIMINARY alpha collection of some "gee whiz" add-ons and improvements to GONZO.13A. Included are...

- a way to measure actual gonzo line widths (!)
- a minor & subtle improvement to the fill justification
- a left keystone justify routine
- a center keystone justify routine
- a right keystone justify routine

%%%%%%%%%%%%%%%%%%%%%%%%%

Keystone justification
is a quite useful tool for
blurbs, advt copy, or point of
purchase sign use, but it has been
extremely difficult to do either quick
or well in the past. Or at least till now.

These Guru gonzo justification routines quickly and easily give you your choice of left, centered, or right keystone justifications in one (or at most two) passes with unbeatably superb print quality. All of the gonzo justification features (multiple line fonts, lots of kerning options, hanging punctuation, space/char ratio control, outstanding typographic quality, etc. etc. are fully retained.

There are three routines called cck, clk, and crk, short for callout centered keystone, callout right keystone, and callout left keystone. Here's how to use cck ...

```
306            % center xposn
490            % y position
true           % true = keystone; false = ordinary justify

(-- electronic tuning diodes --
-- two unusual newsletters --
-- parametric Amplification --
-- association book resources --
-- preventing modem dropouts --
)cck           % your series of strings to be keystoned
```

The algorithm works as follows: An average length of the first two gonzo string lines is taken. An average length of the last two gonzo string lines is taken. From these, a starting width, a slope, and a line increment is chosen. The length of each individual line string is checked, and the starting width is widened as much as is needed to be sure each line fits. Each gonzo line is then shown fill justified at the proper width and position to get the proper keystone effect.

Some hints ...
1. The closer your messages are to a decent keystone ahead of time, the better the final result. Use -false- to check, and then rearrange, reword, or kern as needed.
2. Make sure your initial txtwide is WIDER than the longest string or astoundingly bizarre things will happen.
3. Set sstretch to the MAXIMUM ACCEPTABLE SQUASH for your tightest line.
4. Always change fonts AFTER changing cstretch or sstretch and NEVER before!

For more help:
(520) 428-4073

Gonzo Keystone Justify, Continued...

```
% this is a repeat of the GONZO13B callout width proc ...

gonzo begin

/endtheline {/curwide txtwide roomleft sub def justx cvx exec oktoprint
{printline} if} bind def

/cw {save /snapc1 exch def /oktoadvance false def /oktoprint false def
/linestring linestring2 def /justx (justL) def 3 1 roll /ypos exch def /xpos exch
def stringgonzo curwide snapc1 restore} def
end

%%%%%%%%%%%%%%%%%

% This is a subtle correction to the gonzo fill justify that makes txtwide slightly
more precise ...

gonzo begin

/reallyjustF {swallowandhang roomleft cstretch add numchars 1 sub numspaces
spacecharratio mul add dup 0 eq {pop 0.001} if div dup cstretch add /cfix exch
def spacecharratio mul sstretch add /sfix exch def /xfix 0 def /yfix 0 def} bind
def % unconditional justF

end

%%%%%%%%%%%%%%%

% KEYSTONE STUFF STARTS HERE

gonzo begin

% This is the core keystone code routine. It works by taking an average
% of the first two and last two string lengths to determine the keystone
% slope, increment, and starting width. It then checks the length of
% each line and extends the starting width as needed so everything fits.

/keyproc {/kadj exch def /msg exch def /yy1 exch def /xx1 exch def mark {msg
() search {exch pop exch /msg exch def dup length 0 le {pop} if}{dup length 0 le
{pop} if exit} ifelse} loop ] /karray exch def 0 0 karray 0 get cw 0 0 karray 1 get
cw add 2 div dup 0 0 karray dup length 1 sub get cw 0 0 karray dup length 2
sub get cw add 2 div sub neg karray length 1 sub div dup /kinc exch def 2 div
sub /kstart exch def /txtwide 10000 def 0 0 1 karray length 1 sub {/kpn exch
def 0 0 karray kpn get cw kstart kinc kpn mul add sub 2 copy lt {exch} if pop}
for kstart add  1.01 mul dup /kstart exch def /txtwide exch def /justifylastline
true def 0 1 karray length 1 sub { /posn exch def yy1 xx1 kstart kadj exch posn
karray exch get cf /txtwide txtwide kinc add def /xx1 xx1 kinc kadj def /yy1 yy1
yinc sub def} for} def

% these are the three keystone routines. Use xpos ypos true (msg strings)
% cck, etc. true for keystone; false for ordinary justify.

/cck {save /keysnap exch def exch {{2 div sub} keyproc} {cc} ifelse keysnap
restore} def      % callout centered keystone
/clk {save /keysnap exch def exch {{pop} keyproc} {cl} ifelse keysnap restore} def
% callout left keystone
/crk {save /keysnap exch def exch {{sub} keyproc} {cr} ifelse keysnap restore}
def      % callout right keystone

end  % close gonzo dictionary
```

**Bonus
Supplement
#41-B**

Gonzo
keystone justify

Don Lancaster's PostScript Secrets

Bonus Supplement #41-C

Gonzo
keystone justify

Gonzo Keystone Justify...

```
%  //// demo - remove or alter before reuse. ////

gonzo begin             % open persistently downloaded dictionary

/txtwide 600 def        % set textwide wider than widest line
/yinc 24 def            % pick vertical spacing
/cstretch 0.2 def       % set minimum character excess kerning
/sstretch 0 def         % set minimum space excess kerning

/font1 /Helvetica [16 0 0 16 0 0] gonzofont
/font2 /Helvetica-Oblique [16 0 0 16 0 0] gonzofont

font1                   % select your first font

% the blurb before keystoning ....

306 650 false
(-- electronic tuning diodes --
-- two 2unusual1 newsletters --
-- parametric Amplification --
-- association book resources --
-- preventing modem dropouts --)cck

% a centered keystone ...

306 490 true
(-- electronic tuning diodes --
-- two 2unusual1 newsletters --
-- parametric Amplification --
-- association book resources --
-- preventing modem dropouts --)cck

% a flush left keystone ...

306 330 true
(-- electronic tuning diodes --
-- two 2unusual1 newsletters --
-- parametric Amplification --
-- association book resources --
-- preventing modem dropouts --)clk

% a flush right keystone ...

306 170 true
(-- electronic tuning diodes --
-- two 2unusual1 newsletters --
-- parametric Amplification --
-- association book resources --
-- preventing modem dropouts --)crk

end          % close gonzo dictionary

showpage quit
```

For more help:
(520) 428-4073

Gonzo Keystone Justify...

the blurb before keystoning ....

-- electronic tuning diodes --

-- two *unusual* newsletters --

-- parametric Amplification --

-- association book resources --

-- preventing modem dropouts --

a centered keystone ...

-- electronic tuning diodes --

-- two *unusual* newsletters --

-- parametric Amplification --

-- association book resources --

-- preventing modem dropouts --

a flush left keystone ...

-- electronic tuning diodes --

-- two *unusual* newsletters --

-- parametric Amplification --

-- association book resources --

-- preventing modem dropouts --

a flush right keystone ...

-- electronic tuning diodes --

-- two *unusual* newsletters --

-- parametric Amplification --

-- association book resources --

-- preventing modem dropouts --

Don Lancaster's
PostScript
Secrets

Bonus Supplement #41-D

Gonzo
keystone justify

For more help:
(520) 428-4073

Sometimes parts of a PostScript image will neither appear to print nor generate an error message. Especially with EPS drop ins. The usual cause is printing far off the page or outside of a clipping area. To check, put a "postage stamp" page image in the middle of your sheet by doing a **300 400 translate 0.1 dup scale** at your page start. This shows most out-of-bounds images.

Here's a powerful **array justify** routine that lets you equally space centered messages horizontally or vertically...

```
(32) (31) (30) (29) (28) (27) (26) (25) (24) (23) (22) (21) (20) (19)
     /yspacing 0 def /xspacing 30 def /delimiter ( ) def /showsubstring
(33) {dup stringwidth 2 div pop xposn sub                                (18)
     neg yposn moveto show /xposn xposn xspacing add
     def /yposn yposn yspacing add def} def
(34) /arrayjustify { gsave /msgs exch def translate /xposn 0 def /yposn 0 (17)
     def {msgs delimiter search {exch pop exch /msgs exch def
     showsubstring} {showsubstring exit}
(35) ifelse } loop grestore} def                                         (16)
     % //// demo - remove or alter before reuse. ////
(36) /Helvetica-Bold findfont [9 0 0 9 0 0] makefont setfont             (15)
     100 200 (1 2 3 4 5 6 7 8 9 10 11 12) arrayjustify showpage quit
(1)  (2)  (3)  (4)  (5)  (6)  (7)  (8)  (9)  (10) (11) (12) (13) (14)
```

The most common cause of an **invalidrestore** error is not having a save object on your stack top when you restore. The second most common is leaving a dictionary open. Open dictionaries can get removed with the **end** operator. A new on-stack string can also cause invalidrestore errors.

MATERIAL OF THE MONTH —

Laser print all of your own checks with special magnetic **MICR TONER REFILLS**. Around $20 per bottle from *MICRTECH Group*, 12 Bartal Court, Atco, NJ 08004. (609) 753-8502, and from many other advertisers in *Recharger* magazine (714) 359-8570. Tech info is available from the *MICR Institute* (609) 435-0090, while MICR fonts are available from *The Font Company* at (800) 442-FONT.

PUBLICATION OF THE MONTH —

A free typography newspaper called **TYPEWORLD**. They are especially strong on newer high end desktop publishing product announcements. From *TypeWorld*, 1 Stiles Road, Box 170, Salem NH, 03079. (603) 898-2822.

FREE FONT

Use of precise fine line detail can get tricky at 300 DPI. Here's a fast, simple and sloppy method that uses a "magic" screen...

```
gsave 50 0 {exch pop -.3 lt {1}{0} ifelse} setscreen 0.67 setgray

/f1 {/Helvetica-BoldOblique findfont 40 scalefont setfont} def
/f2 {/Helvetica-BoldOblique findfont 30.2 scale setfont} def

100 300 moveto f1 (F) show
f2 currentpoint 6.5 add moveto (REE  FON) show
f1 currentpoint 6.5 sub moveto (T) show

1 setlinecap 4 setlinewidth 113 300 moveto 1.2 4 rlineto 158 0 rlineto  -1.2
-4 rlineto closepath fill grestore showpage quit
```

Which also shows you how mixed caps can be a useful design tool. Try changing the vertical position and character heights *in quarter point increments* to "fine tune" the line details. Do not scale this free font or move it after fine tuning. Do not change the setgray value.

Don Lancaster's PostScript Secrets

42

TypeWorld

MICR toner

Invalidrestore

Fineline free font

Array justification

For more help:

(602) 428-4073

Don Lancaster's PostScript Secrets

43

Log graphs

Striped font

String exec trick

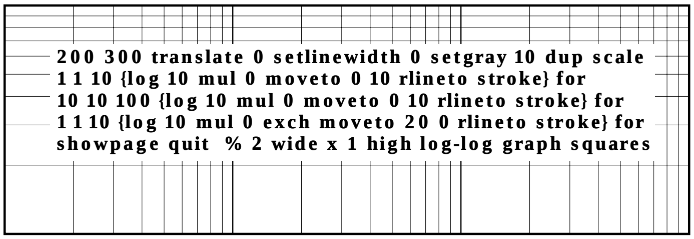
JPEG Tech Specs

9-up *Post-It notes*

Paper printing plates

Word has it that most ordinary paper printing plates can get run on through a LaserWriter just fine. Simply trim the plate to slightly over 8-1/2 inches wide. Open the printer lid after each plate. Process the plate in the usual way and run hundreds of "real ink" copies.

Let's throw a log or two on the fire...



```
200 300 translate 0 setlinewidth 0 setgray 10 dup scale
1 1 10 {log 10 mul 0 moveto 0 10 rlineto stroke} for
10 10 100 {log 10 mul 0 moveto 0 10 rlineto stroke} for
1 1 10 {log 10 mul 0 exch moveto 20 0 rlineto stroke} for
showpage quit  % 2 wide x 1 high log-log graph squares
```

A little known power trick lets you place your *entire* PostScript program inside a string and then execute the string. As in **(mygreat program code) cvx exec.** Uses include preventing stack overflows on complex procs, or bypassing the 6K input buffer to solve comm hassles. The string can be 65K characters long.

MATERIAL OF THE MONTH —

Laser printable **9-UP POST-IT NOTES** from 3-M. Around three cents or so per note. Stock #7709 gives you 25 yellow sheets (225 notes) for $7.89 list. Also available in white and other sizes. From most office supply stores, or *Paper Plus* at (213) 889-9500. From *3-M*, in 3-M Center, St. Paul, MN, 55144. (612) 733-5454.

PUBLICATION OF THE MONTH —

The newest **JPEG TECHNICAL SPECIFICATION REV 8** shows how to dramatically compress your images, especially with PostScript level II. Transmission times and storage space can be reduced by as much as 100:1. Copies of this 122 page "horses mouth" document available from *Adobe Systems* 1585 Charleston Road, Mountain View CA, 94039. (415) 961-4111. Cost is $18.00.

Here's another route to precisely controlling fine line detail at 300 DPI, based on working directly in whole pixels read from an array...

```
/hstripearray [16 4 14 4 12 4 10 4 8 4 7 4 6 4 5 4 4 4 3 4 2 4 2 4 2 3 1 3 1]
def /widestchar 200 def /extrakern 4 def /strr (X) def

/workinpixels {gsave 4.16667 mul floor 0.24 mul exch 4.16667 mul floor
0.24 mul translate 0.24 dup scale} def

/hpixelstripes {/msg exch def workinpixels  0 0 moveto 0 1 msg length 1
sub {msg exch 1 getinterval gsave dup false charpath clip newpath gsave 0
setgray 0 setlinewidth 0 -1 -5 {0 exch moveto widestchar 0 rlineto stroke}
for 0 1 hstripearray length 1 sub {hstripearray exch get {0 0 moveto
widestchar 0 rlineto stroke 0 1 translate 0 0 moveto} repeat 1 currentgray
sub setgray} for grestore grestore stringwidth pop extrakern add 0 translate
0 0 moveto} for grestore} def

%  /// demo - remove or alter before reuse. ///

/AvantGarde-DemiOblique findfont [180 0 0 180 0 0] makefont setfont 50
300 (FREE FONT) hpixelstripes showpage quit
```

The **hstripearray** gives you 16 pixel lines of black, 4 pixel lines of white, 14 of black, 4 of white, etc. The effect usually looks best on letterheads and larger logos and does require some care.

For more help:

(602) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #43-A

Smith chart generator

Smith Chart Generator Code...

These routines explore building a Smith chart and also investigate using Postscript to generate "real" graphs by direct computation. Smith charts have important analytical uses in satellites and microwave electronics.

The Smith chart math is much simpler than it seems at first glance. The real circles are radius 1/(1+r) and center 1/(1+r),0. The reactance circles are radius 1/x and center (1,1/x) and (1,-1/x).

The derivation appears in Section 10.8 of Potter & Fitch THEORY OF NETWORKS AND LINES, Prentice-Hall 1963. The original Smith stuff appears in Electronics v12 Jan 39 p 29-31 and Electronics v17 Jan 44 p140. A newer set of detailed Smith examples appears in Hund MICROWAVE COMMUNICATIONS McGraw Hill 1989. Hewlett-Packard is also very big on Smith Charts, especially on their newer instruments.

Current copyright status of the actual Smith Chart itself is unknown. Emeloid Co of Hillside NJ was the key supplier at one time.

I've purposely left the lettering (easy) and the rotor scale (hard) off as "excercises for the student." Let me know if you really need them. I also didn't bother to completely clean the code up just yet. On the other hand, it ain't broke.

```
%  Some working tools ...
/rr { 1 add 1 exch div dia 2 div mul dup 0 exch 0 360 arc} def  % creates a real part circle path
/xx+ { dup 0 eq {pop 0.001} if 1 exch div dia 2 div mul 0 exch dup 0 360 arc} def % creates a positive imaginary circle path
/xx- { dup 0 eq {0 0 moveto dia 2 div 0 lineto} { 1 exch div dia 2 div mul neg 0 exch dup neg 0 360 arc} ifelse} def  % creates a negative imaginary circle path
```

Discussion: Various regions on the chart have different resolutions. Using irregular PS clipping to isolate these areas would be extremely slow, especially in larger sizes. Instead, starting and ending angles are used. To find these angles, draw a triangle consisting of the circle centers and their intersection. This is a RIGHT triangle because of the orthogonality whose sides are 1/(1+r) and 1/x. Draw a second and COMPLEMENTARY right triangle below it whose arms are the distances from the circle centers to the origin. Play with the math for a while and an angle of 2 atan (x/(1+r)) or its complement pops out.

```
/plotr {/jj exch def /real exch def jj 1 real add atan 2 mul /ang exch def 1 1 real add div dia 2 div mul dup 0 exch ang dup neg arcn} def   % creates a path from the intersection of the negative x to the postive x intercept
/plotr2 {/jj2 exch def /jj1 exch def /real exch def jj1 1 real add atan 2 mul /sang exch def jj2 1 real add atan 2 mul /eang exch def 1 1 real add div dia 2 div mul dup 0 exch sang eang arcn} def  % creates a path from the intersection of the first x to the second x
/plotx {/jj exch def /real2 exch def /real1 exch def jj 1 real1 add atan 2 mul  90 sub /ang1 exch def jj 1 real2 add atan 2 mul 90 sub /ang2 exch def 0 1 jj div dia 2 div mul dup ang1 ang2 arcn} def  % creates a path from the intersection of the first r to the second r
/plotx- {neg /jj exch def /real2 exch def /real1 exch def jj 1 real1 add atan 2 mul 90 sub /ang1 exch def jj 1 real2 add atan 2 mul 90 sub /ang2 exch def 0 1 jj div dia 2 div mul dup ang1 ang2 arc} def
```

two tools to shorten the code

```
/px{3 copy plotx stroke plotx- stroke} def
/px2{3 copy plotr2 stroke neg exch neg plotr2 stroke} def
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

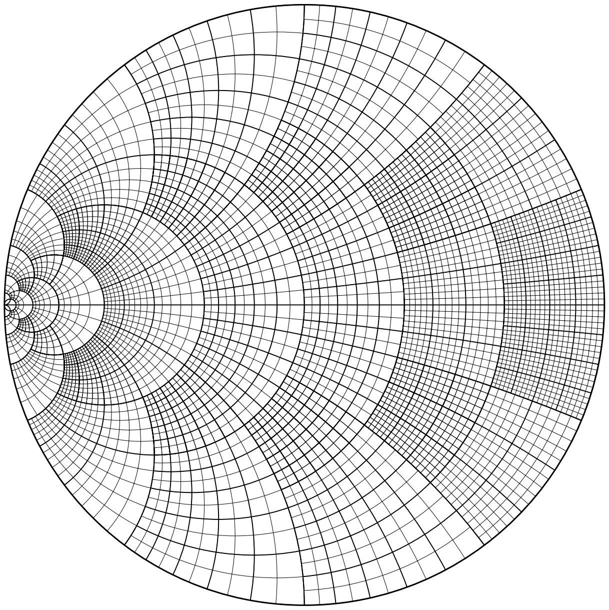
Bonus Supplement #43-B

Smith chart

And the actual chart generator

```
/smithchart {0.6 setlinewidth 0 rr stroke 0.4 setlinewidth 0 0 moveto dia 0 lineto
stroke 0 setlinewidth highresolution {0 0.01 0.2001 {0.2 plotr stroke} for 0.01
0.01 0.2001 {0 exch 0.2 exch px} for 0.2 0.02 0.5001 {0.5 plotr stroke} for 0.02
0.02 0.5001 {0.2 exch 0.5 exch px} for 0 0.02 0.5001 {0.5 0.2 px2} for 0.2 0.02
0.5001 {0 exch 0.5 exch px} for 0.5 0.05 1.001 {1.0 plotr stroke} for 0.05 0.05
1.001 {0.5 exch 1.0 exch px} for 0 0.05 1.001 {1.0 0.5 px2} for 0.5 0.05 1.001 {0
exch 1.0 exch px} for 1 0.1 2.001 {2.0 plotr stroke} for 0.1 0.1 2.001 {1.0 exch 2.0
exch px} for 0 0.1 1.001 {2.0 1.0 px2} for 1 0.1 2.001 {0 exch 2.0 exch px} for 2
0.2 5.001 {5.0 plotr stroke} for 0.2 0.2 5.001 {2.0 exch 5.0 exch px} for 0 0.2
2.001 {5.0 2.0 px2} for 2 0.2 5.001 {0 exch 5.0 exch px} for 5 1 10.001 {10.0 plotr
stroke} for 1 1 10.001 {5.0 exch 10.0 exch px} for 0 0.5 5.001 {10.0 5.0 px2} for 5
1 10.001 {0 exch 10.0 exch px} for 10 2 20.001 {20.0 plotr stroke} for 2 2 20.001
{10.0 exch 20.0 exch px} for 0 2 10.001 {20.0 10.0 px2} for 10 2 20.001 {0 exch
20.0 exch px} for 20 10 50.001 {50.0 plotr stroke} for 10 10 50.001 {20.0 exch
50.0 exch px} for 0 10 20.001 {50 20.0 px2} for 20 10 50.001 {0 exch 50.0 exch
px} for} if  % end hires if used

% low res starts here
0.4 setlinewidth
0 0.05 0.2001 {0.2 plotr stroke} for 0.05 0.05 0.2001 {0 exch 0.2 exch px} for 0.2
0.1 0.5001 {0.5 plotr stroke} for 0.1 0.1 0.5001 {0.2 exch 0.5 exch px} for 0 0.1
0.5001 {0.5 0.2 px2} for 0.2 0.1 0.5001 {0 exch 0.5 exch px} for 0.5 0.1 1.001 {1.0
plotr stroke} for 0.1 0.1 1.001 {0.5 exch 1.0 exch px} for 0 0.1 1.001 {1.0 0.5 px2}
for 0.5 0.1 1.001 {0 exch 1.0 exch px} for 1 0.2 2.001 {2.0 plotr stroke} for 0.2 0.2
2.001 {1.0 exch 2.0 exch px} for 0 0.2 1.001 {2.0 1.0 px2} for 1 0.2 2.001 {0 exch
2.0 exch px} for 2 1 5.001 {5.0 plotr stroke} for 1 1 5.001 {2.0 exch 5.0 exch px}
for 0 1 2.001 {5.0 2.0 px2} for 2 1 5.001 {0 exch 5.0 exch px} for 5 5 10.001 {10.0
plotr stroke} for 5 5 10.001 {5.0 exch 10.0 exch px} for 0 5 5.001 {10.0 5.0 px2}
for 5 5 10.001 {0 exch 10.0 exch px} for 10 10 20.001 {20.0 plotr stroke} for 10
10 20.001 {10.0 exch 20.0 exch px} for 0 10 10.001 {20.0 10.0 px2} for 10 10
20.001 {0 exch 20.0 exch px} for 50 rr stroke 50 50 50.001 {0.0001 exch 2000
exch px} for} def
```

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

50 350 translate           % where?
/dia 500 def              % how big?
/highresolution true def    % lots of detail?
smithchart                % draw one
```



For more help:

(602) 428-4073

Don Lancaster's PostScript Secrets

44

Paper cup font
Bakerizing film
Roman numerals
Calligraphic caps
Paper out indicator
PS: A Visual Approach

Add a stripe of fluorescent bumpersticker material to the bottom of your paper trays. This gives you another "out of paper" indicator that is easier to notice when walking by an unattended printer.

Here's a simple and clean way to merge two PostScript strings...

```
/mergestr {2 copy length exch length add string dup dup 4 3 roll
4 index length exch putinterval 3 1 roll exch 0 exch putinterval} def
% merges top two stack strings into single stack string.
```

To use, just do a **(stringA) (stringB) mergestr** to get **(stringAstringB)**. The proc works by creating a new string of the proper length and then appropriately back-to-back copying the original strings.

And here's how you can use a string merging to generate the first hundred Roman numerals for book boilerplate or whatever...

```
/Romnum {dup 10 div cvi [() (x) (xx) (xxx) (xl) (l) (lx) (lxx) (lxxx) (xc)] exch
get exch 10 mod cvi [() (i) (ii) (iii) (iv) (v) (vi) (vii) (viii) (ix) ] exch get
mergestr} def

% //// demo - remove before use ////

/Helvetica findfont 9 scalefont setfont 100 635 moveto

(The decimal number 18 is )  18 Romnum mergestr
( in Roman Numerals.) mergestr  show showpage quit
```

You can easily extend this technique for higher Roman Numerals, but these are rarely needed as calculated values.

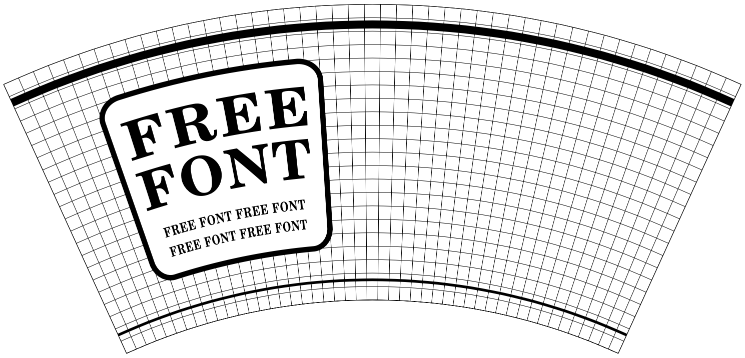
Avoid **ever** using *Zapf* or any other highly calligraphic font as *ALL CAPITALS*. Because of those swashes and flourishes, these fonts should **only** be used in *Mixed Caps* or single letter *M* modes.

MATERIAL OF THE MONTH —

An unusual, cheap, and reusable **BAKERIZING FILM** instantly gives a dense black, a high gloss, and extra durability to any toner image. Great for business cards. Just temporarily place your image in contact with the film and apply heat and pressure. From *Synergetics* Box 809, Thatcher, AZ 85552. (602) 428-4073.

PUBLICATION OF THE MONTH —

Ross Smith's **POSTSCRIPT: A VISUAL APPROACH**, a recommended beginner's book with a focus on grid-based text and graphic constructs. Through *Peachpit Press*, at 1085 Keith Avenue, Berkeley, CA 94708. (415) 527-8555.



While this non-linear
transform is handy for paper cups
or megaphones, the listing and the docs are a tad too long to show here.
See Bonus Supplements 44-A, 44-B, and 44-C for details.

For more help:
(602) 428-4073

Bonus Supplement #44-A

Paper cup
font code

Rootbeer Cup Font Code...

This detailed example shows how to do "rootbeer" or "dixie cup" transforms of *LaserWriter Corner #44* for printing on truncated cones. Shown is PostScript Level I code. The process can be *vastly* simplified and sped-up by going to a Level II machine with its immediately grabbable font paths.

```
%  Copyright c 1992 by Don Lancaster and Synergetics, 3860 West First Street,
%  Box 809, Thatcher AZ 85552. (602) 428-4073. All commercial rights reserved.
%  Personal use permitted so long as this header remains present and intact.

/mychardict 100 dict def mychardict dup  % start a closed dictionary

/F { 722 { 426 0 mt 426 53 li 363 56 li 306 58 306 70 306 123 ct
 306 345 li 316 345 li 383 345 418 326 447 264 ct 458 241 462 215
 464 191 ct 515 191 li 515 537 li 464 537 li 443 439 413 394 306 398 ct
 306 604 li 306 620 304 642 318 654 ct 342 671 374 668 402 668 ct
 548 668 606 593 640 459 ct 690 459 li 683 722 li 23 722 li 23 669 li
 49 669 li 68 668 89 670 107 663 ct 128 657 128 633 128 615 ct
 128 120 li 128 105 128 89 125 74 ct 123 71 122 67 120 64 ct
 118 61 114 59 111 57 ct 101 53 92 53 82 53 ct 22 53 li 22 0 li
 cp}} put dup
/R { 815 { 303 607 mt 303 627 304 653 323 664 ct 346 672 372 671 396 671 ct
 487 671 537 642 537 545 ct 537 437 481 405 383 402 ct 356 401 330
 399 303 400 ct cp 20 722 mt 20 669 li 52 669 li 75 668 100 673 117 656 ct
 129 644 125 593 125 576 ct 125 113 li 123 72 123 55 75 54 ct
 57 53 38 53 20 53 ct 20 0 li 413 0 li 413 53 li 388 53 li
 371 54 353 53 336 56 ct 315 57 309 63 304 83 ct 303 94 303 106 303 117 ct
 303 347 li 335 347 li 350 347 365 347 380 346 ct 475 340 480 280 486 196 ct
 490 136 li 493 97 500 59 528 30 ct 559 -1 598 -15 642 -15 ct
 723 -15 795 18 811 104 ct 816 123 817 143 817 163 ct 767 163 li
 764 131 758 50 707 70 ct 703 74 700 78 698 82 ct 688 102 683 125 678 147 ct
 662 210 li 642 295 594 342 512 361 ct 620 370 727 418 727 542 ct 727
 595 698 651 654 680 ct 602 713 549 719 490 720 ct 415 722 li cp}} put dup
/E { 759 { 696 0 mt 707 265 li 657 269 li 615 124 559 53 401 53 ct
 383 53 365 55 347 57 ct 298 62 300 83 300 127 ct 300 346 li 400 349
 455 289 459 193 ct 513 193 li 509 541 li 458 541 li 447 448 399 391
 300 399 ct 300 588 li 300 605 299 623 306 639 ct
 308 648 315 654 322 659 ct 344 669 369 670 393 670 ct
 553 670 603 608 646 462 ct 696 462 li 689 722 li 16 722 li 16 669 li
 44 669 73 670 100 664 ct 111 662 115 657 119 647 ct
 123 641 122 632 122 625 ct 122 114 li 122 102 123 90 119 79 ct
 110 42 43 56 16 53 ct 16 0 li cp}} put dup
/space { 287 { 287 0 mt}} put dup
/O { 833 { 419 685 mt 578 685 596 477 596 360 ct 596 323 594 283 588 247 ct
 572 154 551 34 431 34 ct 358 34 304 67 275 135 ct 249 197 237 286 237 353
 ct 237 477 249 685 419 685 ct cp 407 -15 mt 513 -15 599 13 677 87 ct
 749 154 794 264 794 361 ct 794 461 755 556 686 628 ct 610 705 520
 737 413 737 ct 319 737 218 698 151 632 ct 76 559 39 461 39 358 ct
 39 264 80 165 144 96 ct 212 23 307 -15 407 -15 ct cp}} put dup
/N { 833 { 304 0 mt 304 53 li 212 55 191 97 191 181 ct 191 553 li 664 -8 li
 726 -8 li 726 512 li 726 541 726 570 732 598 ct 744 654 783 662 833 669 ct
 833 722 li 550 722 li 550 669 li 635 668 661 634 661 552 ct 661 262 li
 272 722 li 19 722 li 19 669 li 62 669 101 650 126 614 ct 126 169 li
 123 82 109 57 16 53 ct 16 0 li cp}} put dup
/T { 722 { 64 451 mt 77 540 124 674 236 674 ct 248 674 269 670 270 655 ct
 271 641 271 626 271 612 ct 271 117 li 271 105 272 91 267 79 ct 266 70
 255 60 247 59 ct 218 56 189 53 159 53 ct 159 0 li 563 0 li 563 53 li
 538 54 512 52 487 55 ct 449 60 449 69 449 105 ct 449 611 li
 449 625 449 640 450 654 ct 451 669 472 673 484 673 ct 597 673 643 544
 657 451 ct 707 451 li 692 722 li 31 722 li 15 451 li cp}} put pop
```

Rootbeer Cup Font Code, continued

Bonus
Supplement
#44-B

Paper cup
font code

```
%  here are the nonlinear operators...mt, li, ct, and cp are nonlinear
% replacements for the usual moveto, lineto, curveto, and closepath operators.
% Note that ONLY these operators are allowed in a starwars transformation.
% special techniques are required for lineto and closepath, since, if only end
% points are used, these operators may take a "short cut" straight across. To
% get around this problem each lineto and closepath is optionally split up into
% numcurvesperlineto pieces. Each piece is replaced with a suitable curved cubic
% spline. The higher the value of numcurvesperlineto, the better the image will
% look but the longer the processing time. A numcurvesperlineto value of 4 is
% recommended for all but the largest lettering, whle 16 should be used for all
% but the very largest of long line graphics.
% A boolean called showlinesascurves decides whether to allow shortcuts or not.
% Using showlinesascurves false greatly speeds up initial layout.

/mt {savecp mark  2 index 2 index ] /lastmt exch  def exch fontsize 0 get mul .001
mul xoffset add exch fontsize 3 get mul .001 mul yoffset add nlt moveto} def /ct
{savecp 3 {6 2 roll exch fontsize 0 get mul .001 mul xoffset add exch fontsize 3
get mul .001 mul yoffset add nlt}  repeat curveto} def
/showlinesascurves false def /numcurvesperlineto 4 def

% here's the switcher that decides what to do with a lineto, the original li allows
% shortcuts, lict does the actual lineto to multiple curveto conversion.

/li {showlinesascurves {lict}{origli} ifelse} def /origli {savecp exch fontsize 0 get
mul .001 mul xoffset add exch fontsize 3 get mul .001 mul yoffset add nlt lineto}
def /lict {/yy exch def /xx exch def yy ycp sub numcurvesperlineto dup add div
/yd exch def xx xcp sub numcurvesperlineto dup add div /xd exch def
numcurvesperlineto
{xcp xd add ycp yd add 2 copy xcp xd 2 mul add ycp yd 2 mul add ct} repeat} def

% the cp closepath operator is similarly steered to avoid shortcuts. The endpoint
% of cp is saved by the last mt moveto in a two element array called lastmt and
% of form [ x y ]. as an internal working tool savecp holds the previous
% UNTRANSFORMED next currentpoint for lict....

/origcp {closepath} def /cp {showlinesascurves {licp}{origcp} ifelse} def /licp {lastmt
aload pop lict closepath } def /savecp {2 copy /ycp exch def /xcp exch def} def

% charproc is what you want to do with each character path. You can get really
% fancy here, doing multiple fills, litho spreads, etc.

/charproc {fill} def % default for a plain old letter

% twshow causes a string to get imaged in the twixtbezier transform. The format
% is -xoffset- -yoffset- (MESSAGE) twshow

/twsshow {/msg exch def /yoffset exch def /xoffset exch def  /strrx (X) def  msg
{strrx exch 0 exch put strrx dup ( ) eq {pop (space)} if cvn mychardict exch get
exec exec charproc fontsize 0 get mul 0.001 mul extrakern add xoffset add
/xoffset
exch def} forall} def /extrakern 0 def % default extra character spacing

% sketchmode lets you map any drawing that consists ONLY of mt, li, ct, and cp
% operators just like you would a character. This operator must be called each
% time IMMEDIATELY before any sketching is done.

/sketchmode {/fontsize [1000 0 0 1000 0 0] def /xoffset 0 def /yoffset 0 def} def

% THE NONLINEAR PAPER CUP TRANSFORM

% In general, any SYNERGETICS nonlinear transform moves each end of each mt
% and li and cp, and each end and each control point of ct by replacing the x
% and y values with a new x' and y' obeying some rule or rules. papercup does a
% nonlinear transform to a vertically oriented segment of a circle ...

/papercup {exch posnadjust 57.2958 mul /ang exch def dup ang sin mul exch ang
cos mul} def /hscaleadjust 1 def /posnadjust {hscaleadjust mul} def
```

Don Lancaster's
LaserWriter
Secrets

Bonus
Supplement
#44-C

Paper cup
font code

Rootbeer Cup Font Code, continued

```
%  //// demo - remove or alter before reuse. ////

% The layout is done on a grid of nominal ten point blocks. You can optionally
% turn this grid on or off.

/pi 3.1415926 def /inches {72 mul} def /cupheight  220 def /cuptopdiameter 180
def /cupbotdiameter 120  def /ch cupheight 10 div def /ctd cuptopdiameter 10
div def /cbd cupbotdiameter 10 div def /circtr ctd dup cbd sub dup 0 eq {pop
0.001} if div ch mul def /circbr circtr ch sub def /ctal ctd circtr div pi mul def
/midgrid {ctal circtr circbr add 2 div  ctal mul floor div} def

% the translation below sets the CENTER POINT of the cup art. this is usually FAR
% BELOW the cup itself and often off page.

/xcen 307 def /ycen 40 def xcen ycen translate 10 dup scale /nlt {papercup} def
/showcupgrid true def /showcupoutline true def /showlinesascurves true def
/numcurvesperlineto 16 def /xoffset 0 def /yoffset 0 def

% put down the center point and cross

gsave 0 setlinewidth -1.5 0 moveto 1.5 0 lineto stroke 0 -1.5 moveto 0 1.5 lineto
stroke grestore

% These routines optionally let you show the grid and the cup outline

/hscaleadjust 1 def gsave sketchmode 0 setlinewidth showcupoutline {ctal 2 div
neg circbr mt ctal 2 div neg circtr li ctal 2 div circtr li ctal 2 div circbr li cp
stroke} if /hscaleadjust midgrid def showcupgrid {circbr 1 circtr { /posn exch def
ctal 2 div hscaleadjust div dup neg posn mt posn li 0 setlinewidth stroke } for
ctal 2 div hscaleadjust div neg 1 ctal 2 div  hscaleadjust div 0.01 add
{/posn exch def posn circbr mt posn circtr li stroke} for} if grestore

% This adjusts the position of the following artwork so that 0 0 is
% in the lower lefthand cup corner

/posnadjust {hscaleadjust mul ctal 2 div sub exch circbr add exch} def


% THE ACTUAL ROOT BEER CUP ART STARTS HERE

% your artwork is done on a simple grid with 0 0 at the lower left. note that ONLY
% mt li ct and cp operators are allowed, because of the nonlinear transformation.
% you can either use only these operators, or else use a font grabber, the
% pathforall operator, or the ADOBE DISTILLERY to do this. First we draw some
% very long horizontal lines...

/numcurvesperlineto 16 def /showlinesascurves true def sketchmode 0.6
setlinewidth 0 20.5 mt 47 20.5 li stroke 0.2 setlinewidth 0 1.5 mt 47 1.5 li stroke

% add a rounded box label graphic, this rounded box has been created with my
% gonzo utilities and then compiled with DISTILL.PS

sketchmode 0.4 setlinewidth /numcurvesperlineto 8 def 12.5 4 mt 6.5 4 li
5.67157 4 5.0 4.67163 5.0 5.5 ct 5 16.5 li 5.0 17.3284 5.67157 18 6.5 18 ct
18.5 18 li 19.3284 18 20 17.3284 20 16.5 ct 20 5.5 li 20 4.67163 19.3284 4 18.5

4 ct cp gsave 1 setgray fill grestore stroke

% Now we put down some text in two different sizes. Note that each character
% used MUST be precaptured as paths to mychardict

/numcurvesperlineto 8 def /charproc {fill} def /extrakern 0.3 def /fontsize [4 0 0 4
0 0] def 6 13.2 (FREE) twsshow /extrakern 0.2 def 5.9 9.3 (FONT) twsshow
/showlinesascurves false def /extrakern 0.05 def /fontsize [0.8 0 0 1 0 0] def 6.7
7 (FREE FONT FREE FONT) twsshow 6.7 5.5 (FREE FONT FREE FONT) twsshow
```

For more help:
(602) 428-4073

Don Lancaster's PostScript Secrets

45

Applique film

Fancy borders

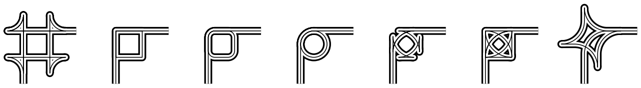
Error reporting

Embossed font

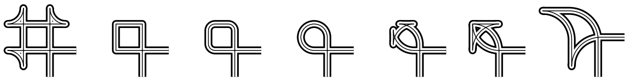
Siggraph Proceedings

PostScript does *not* report errors if results from any internal calculation exceed 10^38 or end up less than 10^-38. Instead, *fixed* values of *Infinity* or *Nan.0* get substituted. This can lead to subtle clipping errors during complex calculations.

A disgustingly serendipitous cheap shot I'll call **ARCTO CROWDING** can give you stunning borders in amazingly few bytes...



```
/path {0 0 moveto 0 100 -20 100 rad arcto 4 {pop} repeat -20 100 -20 80 rad
arcto 4 {pop} repeat -20 80 80 80 rad arcto pop pop pop pop 80 80 lineto}
def

/fancystrokes {gsave 5 setlinewidth 0 setgray stroke grestore gsave 2.5
setlinewidth 1 setgray stroke grestore gsave 0.6 setlinewidth 0 setgray
stroke grestore newpath} def

0 setlinecap 1 setlinejoin gsave  200 300 translate /rad 19.9 def path
fancystrokes grestore showpage quit
```



Upper row has an inside loop. Lower row uses outside loop. Rad values, left to right: -12, 0, 5, 10, 16, 19.9, and 25. Do *not* use a radius exactly the same as your loop, or a *div0* error will result.

This PostScript answer to **CHR$(x)** changes any 0-255 integer into its equivalent single ASCII character string...

```
/chr$ {(X) dup 0 4 -1 roll put} def
```

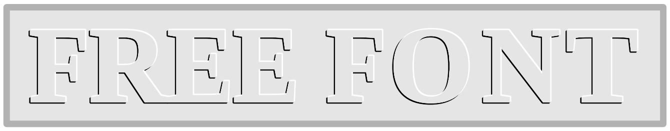
For instance, a decimal 76 on the stack changes into a string (L).

MATERIAL OF THE MONTH —

Two sources for laser printable, transparent stick-on **APPLIQUE FILMS** are *Saga Stickybak*, 400 Hwy 169 South S105, Minneapolis, MN, 55246 (800) 328-0727; and *Stanpat Products* 366 Main, Port Washington, NY 11050 (800) 333-7826.

PUBLICATION OF THE MONTH —

The **SIGGRAPH PROCEEDINGS**, known as annual issue #3 of *Computer Graphics* are full of advanced graphics secrets, algorithms, and ideas. From *ACM*, 7 West 34th Street, New York, NY 10001 (212) 741-4184, or technical libraries.



Here's an "embossed font" that looks really great at 600 DPI...

```
/emboss {gsave /msg exch def /boss exch def translate currentgray gsave 0
setgray 0 0.24 boss {neg dup dup 0.86 mul exch 0.5 mul moveto msg show
dup 0.5 mul exch 0.86 mul moveto msg show} for grestore gsave 1 setgray
0 0.25 boss {dup dup 0.86 mul exch 0.5 mul moveto msg show dup 0.5 mul
exch 0.86 mul moveto msg show } for grestore gsave 0 0 moveto setgray
msg show grestore grestore } def

%  //// demo - remove or alter before reuse ////

155 400 translate 135 25 {dup mul exch dup mul add 1.0 exch sub}
setscreen 0.9 setgray 0 0 moveto 0 54 rlineto 304 0 rlineto 0 -54 rlineto
closepath gsave fill grestore gsave 1 setlinecap 1 setlinejoin 0.7 setgray 5
setlinewidth stroke grestore newpath

/Palatino-Bold findfont [50 0 0 50 0 0] makefont setfont
10 10 0.8 (FREE FONT) emboss grestore showpage quit
```

For more help:

(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #45-A

Serendipitous border contest

PostScript serendipitous border contest...

Just design and upload ANY decent looking raw PostScript border that uses "very few bytes". Your odds of winning Guru books, software, and/or GEnie time are very high. 35 design ideas are included to get you started.

This is a contest with only one rule: Upload an original raw PostScript border that looks good and uses "very few bytes" of code. Prizes will include INCREDIBLE SECRET MONEY MACHINES, other Guru software and books, and free GEnie time. For the very best entry of all, there will be an all expense paid (FOB Thatcher, AZ) tinaja quest for two. Your odds of winning are VERY high.

This is also a great first project in raw PostScript programming.

Some hints appear below. Your particular border does NOT have to use arcto crowding, roundpath, or superstroke. But you might choose to use these and other Guru utilities to keep your total byte count down to a reasonable value. It goes without saying that scanned images need not apply.

Prepare to repel borders.

First, several excerpts from our great #270 GUTIL13A.PTL utilities...

```
%%%%%%%%%% GUTIL13A.PTL EXCERPTS %%%%%%%%%%

% superstroke and superinsidestroke take a predefined path and a top-of-stack
% array of [width1 gray1 width2 gray2 .. widthn grayn] and do multiple strokes
% for wires, fancy borders, or braiding. Note that the FIRST array value pair has to
% be the WIDEST, etc. Use superstroke for wires; superinsidestroke for borders.

/superstroke { save /sssnap exch def /sscmd exch def mark 0 2 sscmd length 2
div cvi 1 sub 2 mul {/aposn exch def gsave sscmd aposn get setlinewidth sscmd
aposn 1 add get setgray stroke grestore} for cleartomark sssnap restore newpath}
def

/flushends {0 setlinecap} def        % rounded path ends
/roundjoins {1 setlinejoin} def      % rounded path joins

% This creates a rounded path from -radius- [x1 y1  x2 y2 ... xn yn] roundpath.
% Does NOT round path ends.

/roundpath {/rpdata exch def /rprad exch def rpdata length 1 sub cvi /rppoints
exch def rpdata 0 get rpdata 1 get moveto 2 2 rppoints 2 sub {/rpvalue exch def 0
1 3 {rpdata exch rpvalue add get } for rprad arcto pop pop pop pop} for rpdata
rppoints 1 sub get rpdata rppoints get lineto} def

%%%%%%%%%% END GUTIL13A.PTL EXCERPTS %%%%%%%%%%
```

```
SOME HINTS: Here are more examples of some serendipitous border code...
/ss {[5 0 2.5 1 0.6 0] superstroke} def  flushends roundjoins

% Top row has OUTSIDE TRIANGLE

/path { [0 70 0 120 -20 100 30 100 ] roundpath} def

gsave  80 0 translate  -6 path ss grestore
gsave 150 0 translate   0 path ss grestore
gsave 220 0 translate   5 path ss grestore
gsave 290 0 translate   8 path ss grestore
gsave 360 0 translate  10 path ss grestore
gsave 430 0 translate  11 path ss grestore
gsave 505 0 translate  16 path ss grestore

% Second row has FLAT INSIDE NOTCH

/path { [0 50 0 80 20 80 20 100 60 100 ] roundpath} def

gsave  80 0 translate -12 path ss grestore
gsave 150 0 translate   0 path ss grestore
gsave 220 0 translate   5 path ss grestore
gsave 290 0 translate  10 path ss grestore
gsave 360 0 translate 15.8 path ss grestore
gsave 430 0 translate 19.9 path ss grestore
gsave 505 0 translate  25 path ss grestore
```

For more help:
(520) 428-4073

PostScript serendipitous border contest...

```
% Third row has ANGLED INSIDE NOTCH

/path { [0 50 0 80 30 70 20 100 50 100 ] roundpath} def

gsave  80 0 translate  -10 path ss grestore
gsave 150 0 translate    0 path ss grestore
gsave 220 0 translate    5 path ss grestore
gsave 290 0 translate    8 path ss grestore
gsave 360 0 translate   15 path ss grestore
gsave 430 0 translate 19.9 path ss grestore

/path { [0 40 0 80 30 70 20 100 60 100 ] roundpath} def
gsave 505 0 translate   25 path ss grestore

% Fourth row has INSIDE LOOP

/path {[0 60 0 100 20 100 20 80 0 80 0 100 40 100] roundpath} def

gsave  80 0 translate  -12 path ss grestore
gsave 150 0 translate    0 path ss grestore
gsave 220 0 translate    5 path ss grestore
gsave 290 0 translate   10 path ss grestore
gsave 360 0 translate 15.8 path ss grestore
gsave 430 0 translate 19.9 path ss grestore

/path {[0 60 0 100 10 100 10 90 0 90 0 100 40 100] roundpath} def
gsave 505 0 translate 17.5 path ss grestore

% Fifth row has OUTSIDE LOOP

/path { [0 60 0 100 -20 100 -20 80 20 80 ] roundpath} def

gsave  80 0 translate  -12 path ss grestore
gsave 150 0 translate    0 path ss grestore
gsave 220 0 translate    5 path ss grestore
gsave 290 0 translate   10 path ss grestore
gsave 360 0 translate 15.5 path ss grestore
gsave 420 0 translate 19.9 path ss grestore

/path { [0 50 0 100 -20 100 -20 80 20 80 ] roundpath} def
gsave 495 10 translate  22 path ss grestore

% Sixth row has OUTSIDE TRIANGLE

/path { [0 60 0 90 20 90 30 70 10 80 10 100 40 100 ] roundpath} def

gsave  80 0 translate   -10 path ss grestore
gsave 150 0 translate     0 path ss grestore
gsave 220 0 translate     5 path ss grestore
gsave 290 0 translate     8 path ss grestore
gsave 360 0 translate   9.9 path ss grestore
gsave 430 0 translate    11 path ss grestore
gsave 505 0 translate    16 path ss grestore

% Seventh row has OFFSET INSIDE LOOP

/path { [0 60 0 90 20 90 20 80 10 80 10 100 40 100 ] roundpath} def

gsave  80 0 translate   -10 path ss grestore
gsave 150 0 translate     0 path ss grestore
gsave 220 0 translate     5 path ss grestore
gsave 290 0 translate     8 path ss grestore
gsave 360 0 translate   9.9 path ss grestore
gsave 430 0 translate    11 path ss grestore
gsave 505 0 translate    16 path ss grestore
```

Don Lancaster's PostScript Secrets

Bonus Supplement #45-B

Serendipitous
border contest

For more help:
(520) 428-4073

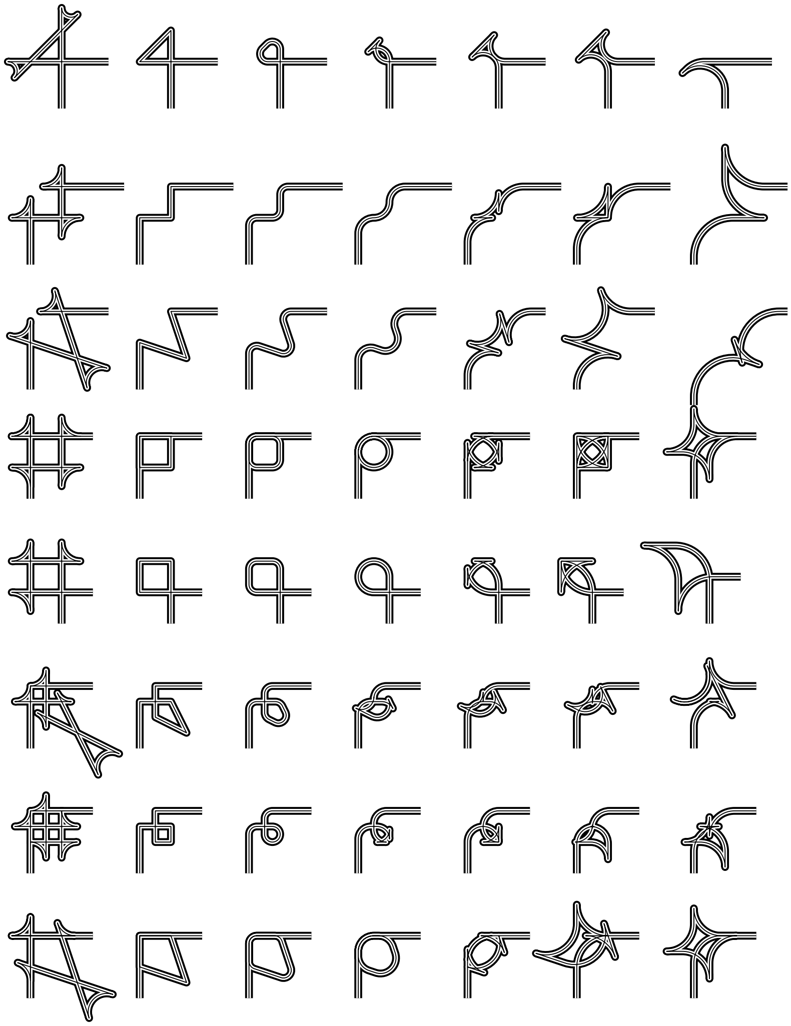
Don Lancaster's PostScript Secrets

Bonus Supplement #45-C

Serendipitous border contest

PostScript serendipitous border contest...

```
% Bottom row has INSIDE TRAPAZOID

/path {[0 60 0 100 20 100 30 70 0 80 0 100 40 100] roundpath} def

gsave  80 0 translate  -12 path ss grestore
gsave 150 0 translate    0 path ss grestore
gsave 220 0 translate    5 path ss grestore
gsave 290 0 translate   10 path ss grestore
gsave 360 0 translate 15.8 path ss grestore
gsave 430 0 translate 19.9 path ss grestore

/path {[0 60 0 100 10 100 10 90 0 90 0 100 40 100] roundpath} def
gsave 505 0 translate 17.5 path ss grestore

grestore showpage
```



For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

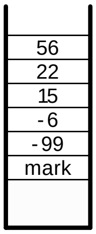
46

Hygrometers
A shadow font
Fast insertion sort
Purple Book revised
Shortening filelengths

A small dab of beeswax on the tip of your *Phillips* screwdriver should greatly simplify most routine repair work on your PostScript printer. The screw stays on the driver until you tighten or remove it.

Here's a fast insertion sort that puts numbers in stack order...

```
/insort {counttomark dup 3 lt {2 eq {2 copy gt {exch} if} if}{1 sub
/#values exch def /newvalue exch def #values dup 2 div cvi dup
2 add index newvalue lt {exch} if pop /#values exch def 0
#values -1 0 { index newvalue lt {1 add}{exit} ifelse} for #values
sub neg 1 add newvalue exch 1 roll} ifelse} bind def

% //// demo - remove or alter before reuse. ////

mark 45 insort -99 insort 22 insort -6 insort 66 insort 15 insort ]
/mygreatvalues exch def

mygreatvalues == flush  quit
```



As with typical stack operations, there is a Level I limit around 500 values. For longer sorts, a tree algorithm is faster and more appropriate.

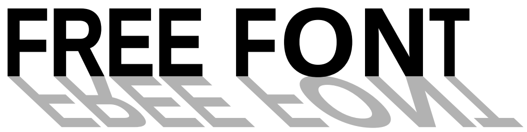
Your PostScript filelengths can be significantly shortened by rounding the *final* positioning numbers to *four* decimal places. The accuracy is still good enough, even at medium-high resolutions. By running at a suitable subscale, you can eliminate the decimal points from most numbers, saving even more bytes.

MATERIAL OF THE MONTH —

To minimize paper curling and jams, keep your printer humidity in the 20 to 45 percent range. Your best source for several ultra-cheap **$5 HYGROMETERS** is *Klockit*, Box 636, Lake Geneva, WI 53147. Call them at (800) 566-2548. Sources for more expensive hygrometers do include *Abbeon-Cal* at (805) 966-0810, *Edmund Scientific* found at (609) 573-6250, and *Heathkit* at (616) 982-3200.

PUBLICATION OF THE MONTH —

David Holtzgang's **PURPLE BOOK**, **UNDERSTANDING POSTSCRIPT PROGRAMMING** is another beginner's book on PostScript. It is particularly good on creating and modifying fonts. Recently revised. *Sybex*, 2021 Challenger Drive #100, Alameda, CA 94501. (800) 227-2346. One stocking source is *Synergetics* found at (602) 428-4073. Also available via *GEnie* PSRT email.



Nuff fancy stuff. Here's a simple shadow font...

```
200 100 translate
106 45 {dup mul exch dup mul add 1.0 exch sub} setscreen

0.8 setgray
/AvantGarde-Demi findfont [40 0 32 -30 0 0] makefont setfont
0 0 moveto (FREE FONT) show  % the shadow

0 setgray /AvantGarde-Demi findfont [40 0 0 40 0 0] makefont setfont 0 0
moveto (FREE FONT) show  % the message

showpage quit
```

If your font array is **[a b c d e f]**, then **a** sets the width, **b** the climb, **c** the lean, **d** the height, **e** the xshift, and **f** the yshift. A negative **d** prints upside down. The negative **a** prints backwards.

Always use a non-putrid gray and *print the shadow first*. This eases problems on characters that go slightly below the baseline. Avoid all lower case descenders.

Fancier versions of this code can set a shadow angle and then use trig to automatically pick the right font matrix for you.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #46-A

Insertion sorts

PostScript insertion sorts...

An insertion sort is usually MUCH faster than a classic bubble sort. Values are placed one at a time into an already presorted array.

Let's call the pile of marked and presorted numbers on the stack a HEAP. Here is one algorithm for a true insertion sort...

```
(1) Find the HALF of the heap the new value belongs in.
(2) Find the QUARTER of the heap the new value belongs in.
(3) Find the EIGHTH of the heap the new value belongs in.

(n) Repeat this process until you know exactly where the
    value belongs. n times for values less than 2^n.

(n+1) Roll over the needed values to do the correct insertion.
```

A true insertion sort theoretically takes (n)log(n) time units to sort, compared to the ridiculously longer (n)^2 of a bubble sort. Insertions sorts are theoretically among the fastest possible.

Unfortunately, overhead can cut into the efficiency of a true insertion sort, especially when working with fewer values. For instance, a dozen "easy" comparisons inside a loop might execute considerably faster than one "hard" comparison outside of a loop.

Two working insertion sorts are shown below that seem to be a useful compromise between overhead and efficiency. While they certainly can be further improved, they are fast, useful, and understandable as is.

Here is the current algorithm used...

```
(1) If there are zero values presorted, insert the value.

(2) If there is only one value presorted, compare values
    and exchange if needed.

(3) Do the following steps ONLY if there are two or more
    presorted values...

    (a) Compare the new value agains the MIDDLE value of
        the heap.

    (b) Compare the new value against sequential values
        in the heap, starting either at the bottom or in
        the middle, per the results of (a).

    (c) Roll the new value into the appropriate heap position.
```

The PostScript stack only allows 500 or so values maximum, so this method in this form is only useful to sort 500 or fewer values. It can be easily extended to sort any length.

The utilities shown below can certainly be improved and fine tuned. I stopped here because they were "good enough' for what I wanted.

A free INCREDIBLE SECRET MONEY MACHINE for any faster code.

For more help:
(520) 428-4073

PostScript insertion sorts...

Don Lancaster's PostScript Secrets

Bonus Supplement #46-B

Insertion sorts

INSERTION SORT UTILITIES

```
% /insort does an insertion sort, placing the top stack object in the appropriate
position in a mark delimited set of stack values. The LOWEST value is just above
the mark, as in  mark -6 -4 0 2 73.

/insort { counttomark dup 3 lt {2 eq {2 copy gt {exch}if}if } {1 sub /numvalues exch
def /newvalue exch def numvalues dup 2 div cvi dup 2 add index newvalue lt
{exch} if pop /numvalues exch def 0 numvalues -1 0 { index newvalue lt {1
add}{exit} ifelse} for numvalues sub neg 1 add newvalue exch 1 roll} ifelse} bind
def

% /abinsort does an absolute insertion sort, placing the top stack object in the
appropriate position in a mark delimited set of stack values. The LOWEST
ABSOLUTE value is just above the mark: mark 0 2 -4 -6 73.

/abinsort { counttomark dup 3 lt {2 eq {2 copy abs exch abs exch gt {exch}if}if}{1
sub /numvalues exch def /newvalue exch def numvalues dup 2 div cvi dup 2 add
index abs newvalue abs lt {exch} if pop /numvalues exch def 0 numvalues -1 0 {
index abs newvalue abs lt {1 add}{exit} ifelse} for numvalues sub neg 1 add
newvalue exch 1 roll} ifelse} bind def

%  //// demo - remove or alter before reuse. ////

1200 {37 sin pop} repeat        % optional host delay

% simple insertion sort example...

mark

45  insort
-99 insort
22  insort
-6  insort
15  insort
]

/mygreatvalues exch def
mygreatvalues == flush

% simple absolute sorting example

mark
45  abinsort
-99 abinsort
22  abinsort
-6  abinsort
15  abinsort
]

/mygreatabsvalues exch def
mygreatabsvalues == flush

%  sorting an existing array...

/unsortedarray [ 23 -56 73 41 -98 -22 85 34 -2 -2 6 71] def

mark
unsortedarray {insort} forall
]

/sortedresult exch def
sortedresult == flush

quit
```

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

47

Black paper
Black stripe fix
Binding systems
Auto-tracking artwork
Software Developer Kit

Don't overlook *black* paper and cover stock for special effects. Use Kroy Color for impressive business cards. Or even try a super subtle black-on-black for pricey and upscale ultra-yuppy effects. Two good sources: *Paper Plus* (800) 272-7377 or *Paper Direct* (800) A-PAPERS.

PostScript strings can inexplicably change all their contents halfway through a long program if you are not extra careful. Ferinstance...

```
/currentstring (ABC) def
/oldstring currentstring def
```

does **not** create a second string named **oldstring**. All it does is stash a pointer right back to *currentstring*. If *currentstring* changes with a *put* or *putinterval*, so does *oldstring*. But...

```
/currentstring (ABC) def
/oldstring currentstring dup length string cvs def
```

safely *de-references* itself by creating a brand new *oldstring* having no pointer back to the original. That *cvs* creates the new string.

That horrible black rightside stripe that often shows up on your SX cartridge output can sometimes get cured by thoroughly cleaning the cartridge corona wire and then placing two strip magnets made from magnetic sign sheeting inside the corona assembly. You could try either *Recharger* (714) 359-8570 or *Signcraft* (813) 939-4644 magazines for suitable magnetic sheeting suppliers.

MATERIALS OF THE MONTH —

Two newer **BINDING SYSTEMS** that I find to be way overpriced but offer fresh approaches to your Book-on-demand publishing: The **CHANNELBIND** system which simply clamps pages in place with no punching. Their hard covers look very good. From *Channelbind*, 3 Townline Circle, Rochester NY 14623. (800) 562-7188. And the brand new **OPTITHERM** system which uses a revolutionary peel-and-strip hot glue system, allows any cover materials, variable thickness, and easily lets you use any spline lettering. From *Planax*, 15 East 26th Street, Suite 1908, New York, NY, 10010 (212) 532-1988.

PUBLICATION OF THE MONTH —

Adobe's new **SOFTWARE DEVELOPMENT KIT** is a big box full of all sorts of useful goodies. Red, blue, green, and black books. Thick specification, programming and documentation notebooks. Lots of IBM and Mac software, fonts, Level II support, and more. $250 to developers, $500 to others. From *Adobe Systems*, 1585 Charleston Road, Mountain View, CA 94039, (415) 961-4400.

FREE FONT

Making your artwork auto-track *awidthshow* strings having variable character and space kerning can get tricky...

```
/autoart {gsave /str exch def currentpoint pop /xstart exch def spacekern 0
32 charkern 0 str awidthshow 1 setlinecap currentpoint dropline sub exch
pop xstart underrun sub exch moveto str stringwidth pop str length 1 sub
charkern mul add 0 str {32 eq {1 add}if} forall spacekern mul add underrun
add overrun add 0 rlineto linethick setlinewidth stroke grestore } def

% //// demo - remove or alter before reuse ////

/NewCenturySchlbk-BoldItalic findfont [25 0 0 25 0 0] makefont setfont

/charkern 1 def /spacekern 4 def /overrun 2 def
/underrun 7 def /dropline 6 def /linethick 4 def

100 200 moveto (FREE FONT) autoart  showpage quit
```

The same concept can be used for nearly any text-surrounding graphic. My *superstroke* and *superinsidestroke* utilities (From *LaserWriter Secrets #26*) sure can spruce up your headers in a resume or whatever.

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

Bonus Supplement #47

Classic bubble sort

Classic bubble array sorts...

While very klutzy and crude, the classic bubble sort can often be the "best" method for sorting a few to a few dozen values in an array. The algorithm is obvious. Start with the final array value. Compare it against the previous value, and exchange the values if the final value is larger, thus "bubbling" largest values to the left.

On the first pass, the largest value will end up at the left. You then repeat the process for all the remaining UNSORTED values. To sort n items requires n + (n-1) + (n-2) + ... comparisons, or (n)(n+1)/2 or roughly (n^2)/2 comparisons. To reverse the final order of either version, change lt to gt.

THE BUBBLESORT UTILITIES:

```
% bubblesort accepts an array of up to 500 elements on the stack top and does a
% classic bubble sort, replacing it with a similar array whose LARGEST value is
% first and SMALLEST value is last.

/bubblesort {mark exch aload pop counttomark /idx exch def { 0 1 idx 1 sub {pop
2 copy lt {exch} if idx 1 roll} for idx 1 roll /idx idx 1 sub def idx 0 eq {exit} if} loop
] } def

% bubbleabssort accepts an array of up to 500 elements on the stack top and
% does a classic bubble sort, replacing it with a similar array whose LARGEST
% absolute value is first and left, while the SMALLEST absolute value is last and
% to the right. Thus larger positive or negative numbers come earlier than smaller
% positive or negative numbers.

/bubbleabssort{mark exch aload pop counttomark /idx exch def { 0 1 idx 1 sub
{pop 2 copy abs exch abs exch lt {exch} if idx 1 roll} for idx 1 roll /idx idx 1 sub
def idx 0 eq {exit} if} loop ] } def
```

NOTE: Two way comm REQUIRED for this demo to run. Be ABSOLUTELY CERTAIN to use Serial, AppleTalk, or Shared SCSI comm.

```
%  //// demo - remove or alter before reuse ////

1000 {37 sin pop} repeat  % optional response delay
([ 1 2 -5 3 9 4 -6 5 7 6 8 -9 7 5 4 3 3 -4 2] bubblesort provides \r) print
[ 1 2 -5 3 9 4 -6 5 7 6 8 -9 7 5 4 3 3 -4 2] bubblesort ==
(\r) print
([ 1 2 -5 3 9 4 -6 5 7 6 8 -9 7 5 4 3 3 -4 2] bubbleabssort provides \r) print
[ 1 2 -5 3 9 4 -6 5 7 6 8 -9 7 5 4 3 3 -4 2] bubbleabssort  ==
(\r) print flush
```

A REALLY OFF-THE-WALL EXAMPLE:

```
% Accumulating the sums of products is often used in solving linear equations
% and in digital signal processing. If, in the process of accumulation, large values
% are added to smaller ones, the accuracy can decrease, leading to subtle but
% serious errors during advanced PostScript work.

% For instance, summing these numbers IN THIS ORDER with PostScript...
%  4500.1 - 4502.1 + 43 - 43 + 15.3 - 9.9
%   + 3.3 - 6 +  1.3 + 0.0042 + 0.0052  = 2.0094 (The CORRECT result)

% But summing these SAME numbers IN THIS ORDER with PostScript...
%  4500.1 + 0.0042 -4502.1 + 0.0052 + 1.3
%  -9.9 + 3.3 + 15.3 - 6 + 43 - 43  = 2.00928 (The WRONG result)

% In the second case, accuracy is lost because of roundoffs that occur when very
% small values are subtracted from very large values.
% To get the best possible results in any PostScript accumulation, presort your
% array by absolute values, so that the LARGEST values are added to each other
% first, followed by progressively SMALLER values.

% For instance...
% [ -4502.1 0.0042 4500.1 0.0052 1.3 -9.9 3.3 15.3 -6 43 -43 ] bubbleabssort
% sorts to [ -4502.1 4500.1 43 -43 15.3 -6 -9.9 3.3 1.3 0.0052 0.0042]
% Summing the first array gives a WRONG answer, while summing the presorted
% array gives you the RIGHT answer.

quit
```

For more help:
(520) 428-4073

Don Lancaster's PostScript Secrets

48

Vertical font

Printer's Devil

Custom kerning

Custom envelopes

Undefinedfilename

Toner thermography

Addressing your envelopes *first* and then manufacturing them *last* is one idea whose time is long overdue. The obvious advantages here include an unlimited materials choice, fast and jam-free unattended flat-sheet feeding, letter perfect print quality, and very few thermal fuser cycled lickum-stickum surprises. New products such as that roll-on *DryLine Permanent Adhesive* from *Liquid Paper* can greatly ease assembly.

Here's a quick-and-dirty custom kerning routine that works on any font and is especially good on headlines...

AWARD
Cañon
RESET

```
/hkern {currentpoint 3 1 roll sub neg exch moveto} def

% /// demo - remove or alter before reuse ////

/Helvetica findfont [30 0 0 30 0 0] makefont setfont

100 500 moveto
(A) show 3 hkern (W) show 3 hkern (ARD) show
showpage
```

Besides tightening awkward character combinations, non-obvious uses include complement bars and foreign symbol overlays. This similar routine can be used for vertical kerning...

```
/vkern {currentpoint 3 -1 roll sub moveto} def
```

Another ultra-powerful kerning trick is to *pre-define* two custom characters in a font that only do a +1 or -1 point kern. This lets you custom adjust circular text, or use other fancy routines where the kerning was previously unavailable.

The latest versions of Adobe's hard disk firmware have stupidly locked out all the /FC and /BM font cache files, and return an *undefinedfilename* error if you try to read them. To access these files, use shared SCSI comm. Or else any SCSI system with a read-and-write-to-sector capability. The IIe SCSI card is ideal for this. But similar solutions exist for all major hosts.

MATERIAL OF THE MONTH —

A great breakthrough: **REAL TONER-COMPATIBLE THERMOGRAPHY**. Just print as normal. Spray on glop "A" to soften and expand the toner. Dust on powder "B". Then apply heat from desklamp "C". For raised letterheads, business cards, and even Braille. Lots of gold, silver, colors and metallics. From *Bennet LaserBrite* at 720 Fourth Street Southwest, Rochester, MN 55902. (507) 280-9101.

PUBLICATION OF THE MONTH —

The **PRINTER'S DEVIL**, a labor-of-love newsletter for small scale press owners and publishers. "A press in every home. A home in every press." $5 per year subscription. *The Printer's Devil*, Mother of Ashes Press, Harrison, ID 83833. Their phone number is (208) 689-3738.

FREE FONT

How do you show a little vertical lettering on any otherwise horizontal layout? By far the fastest and easiest method is to *predefine* a vertically-printing font. Instead of the stock **[wide 0 0 high 0 0]** font matrix, you can just use **[0 high -wide 0 0 0]** instead...

```
/Bookman-Demi findfont [0 25 -20 0 0 0] makefont
setfont

300 400 moveto
(FREE FONT) show
showpage
```

Do not forget that minus sign! On the *ashow*, *wshow*, or *awidthshow* procs, you interchange the x,y increment values. For custom kerning, you can use the *vkern* proc shown above.

FREE FONT

For more help:

(520) 428-4073

Bonus Supplement #48-A

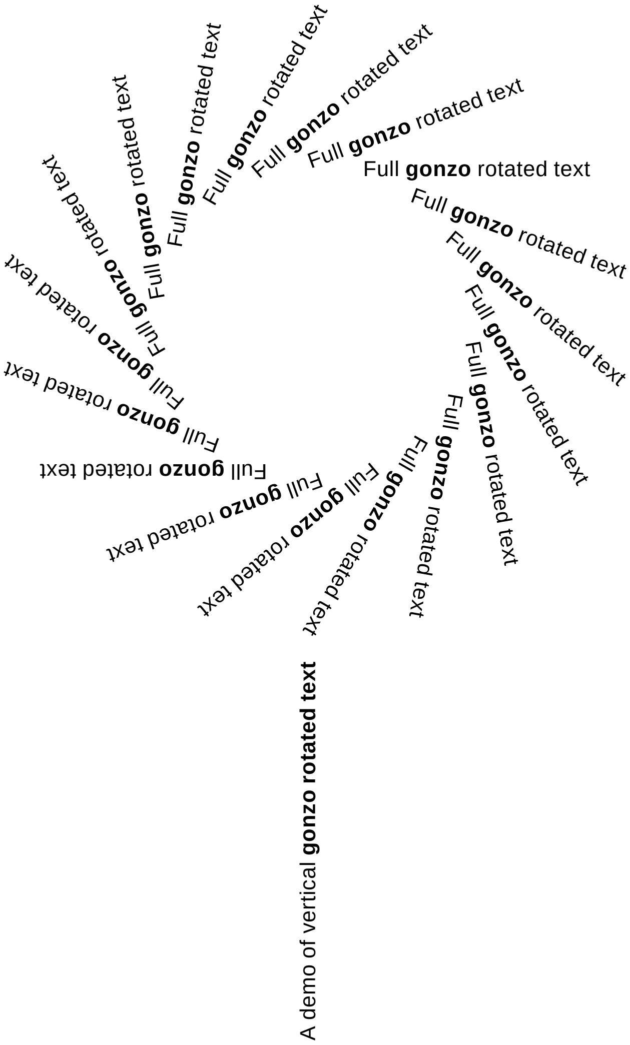
Rotating Text

Rotating Text...

```
% (B) angle justify 2.20.91

% allows callout text to run at any angle. Format is
% xpos ypos angle (message) cca, etc ...


% CALLOUT JUSTIFY MODES

% cl accepts an input of form xpos ypos (message) cl and shows it
% at xpos left and within xpos + textwide right.

/cl {save /snapcl exch def /linestring linestring2 def /justx (justL)
def 3 1 roll /ypos exch def /xpos exch def stringgonzo snapcl restore} def

% cf accepts an input of form xpos ypos (message) cf and shows it
% flush left at xpos and flush right at xpos + textwide.

/cf {save /snapcf exch def /linestring linestring2 def /justx (justF)
def 3 1 roll /ypos exch def /xpos exch def stringgonzo snapcf restore} def

% cc accepts an input of form xpos ypos (message) cc and centers it
% on xpos. txtwide IS IGNORED, AND ANY WIDTH WILL GET CENTERED.

/cc {save /snapcc exch def /linestring linestring2 def /txtwide 5000
def /justx (justC) def /pmrun 0 def 3 1 roll /ypos exch def 2500
sub /xpos exch def stringgonzo snapcc restore} def

% cr accepts an input of form xpos ypos (message) cr and sets it
% flush right against xpos. txtwide IS IGNORED, AND ANY WIDTH
% WILL SET FLUSH RIGHT.

/cr {save /snapcm exch def /linestring linestring2 def /txtwide 5000
def /justx (justR) def /pmrun 0 def 3 1 roll /ypos exch def 5000
sub /xpos exch def stringgonzo snapcm restore} def

gonzo begin

/rotstuff {gsave 4 2 roll translate exch rotate 0 exch 0 exch} def

/cca {rotstuff cc grestore} def   % center
/cla {rotstuff cl grestore} def   % left
/cfa {rotstuff cf grestore} def   % fill
/cra {rotstuff cr grestore} def   % right
end
```

Rotating Text...

```
% demo

gonzo begin
250 100 90 (A demo of vertical gonzo rotated text) cla
gsave 250 400 translate
18 {20 60 0 (Full gonzo rotated text) cla 20 rotate} repeat
grestore end

showpage quit

% .txt changes cca cla cfa rotstuff with examples
```


Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
Full **gonzo** rotated text
A demo of vertical **gonzo rotated text**

Don Lancaster's PostScript Secrets

Bonus Supplement #48-B

Rotating Text

For more help:
(520) 428-4073