Typesetting on Personal Computers

"Free" TEX Software for IBM PCs

Jon Radel

Since there have recently been several confusing mentions of the disk copying service I supply to the TEX community, I would like to take this opportunity to clarify matters a bit and call attention to my service for those people who missed those mentions. I make copies of a variety of material of use for running TEX on an IBM PC or clone. The charge is nominal — to cover my expenses in gathering the material — if you supply the floppies and a return mailer. I can also supply the disks if you prefer to simply send money. I have, at the moment, two ports of TEX itself, one of METAFONT, Nelson Beebe's DVI drivers as well as some other drivers and previewers, the LATEX-style collection, back issues of TEXhax and TEXMAG, and a variety of other interesting material. I make an effort to carry the most recent version of programs, but I can make no guarantees as I am in part dependent on the authors to let me know about new versions.

I would prefer that you send all mail about this software to me at Jon Radel, P. O. Box 2276, Reston, VA 22090. To get the details on ordering, and the current list of what I have, please send a self-addressed envelope. Attach 45 cents postage in the U.S. Outside the U.S., send International Reply Coupons, 2 for Canada and Mexico, 4 for elsewhere, or, if more convenient for you, US$2.25.

Incidentally, if you have created any software of use to someone using TEX on an IBM PC, I would be most interested in hearing about it if you are willing to give me permission to distribute it.

◇ Jon Radel
P. O. Box 2276
Reston, VA 22090
`jonradel@icecream.princeton.edu`

Public Domain TEX for the Mac

Andrew Trevorrow

OzTEX 1.0 is a public domain version of TEX for the Macintosh. It aims to provide a standard TEX environment that can be easily extended or customized. People with access to TEX on some other computer should feel right at home using OzTEX, particularly those who use PSPRINT and DVItoVDU on a VAX/VMS or UNIX mainframe.

A brief description

Here's a quick look at OzTEX's major features:

- The complete distribution requires ten 800K disks. Five of these are full of PK files (for a 300dpi write-black laser printer such as the Apple LaserWriter). Another two disks contain the entire source code. OzTEX is written in Modula-2 under MPW (Macintosh Programmer's Workshop).
- The OzTEX application includes TEX (actually INITEX so users can create their own format files), a DVI page previewer and a PostScript driver that can send output to the current printer or to a text file.
- The three most popular formats are supplied: Plain, LATEX and AMS-TEX.
- OzTEX reads standard TFM and PK files and reads and writes standard DVI files.
- The previewer can cope with just about any DVI file you're ever likely to create, including those generated by another TEX system. Have you ever wondered what `trip.dvi` (the DVI file created by Knuth's trip test) looks like?
- The application includes a Help menu which you can easily extend or modify.
- A configuration file is read when starting up and controls much of OzTEX's default behaviour. This simple text file can be edited to suit your particular requirements. Some of the parameters you can specify include the printer resolution, the paper dimensions, a list of the formats that appear in the TEX menu, and a list of all TFM file names for printer-resident PostScript fonts.
- A 22-page user guide is supplied, including its LATEX source. By the time you read this article I should also have finished a system guide aimed at programmers who'd like to modify OzTEX.

It's not all good news however. There is still plenty of room for improvement:

- There is no integrated text editor. OzTEX is distributed with ΣEdit, a public domain DA editor written by Leonard Rosenthol.
- OzTEX requires a PostScript printer.
- `\special` handling is fairly unsophisticated. OzTEX allows the inclusion of a PostScript file along with optional code prefixed to the file. There is currently no support for previewing PICT or EPSF files.
- Previewing DVI pages is not as fast as I'd like, particularly on a Mac Plus.

Future development of OzTEX is likely but will occur at a fairly sedate pace unless I can find people prepared to help with the programming or provide financial support. Send your bug reports, comments and offers of help to the address shown at the end of this article.

Where to get OzTEX

The following people have volunteered to help distribute OzTEX. Please get in touch with the person nearest you. By the time you read this article it is likely that OzTEX will also be available electronically from various Mac archive sites. People without access to email should try their local Mac user group.

In Australia and New Zealand:

`addie@rhea.trl.oz`	Ron Addie, Melbourne
`keady@madvax.uwa.oz`	Grant Keady, Perth
`rks105@phys6.anu.oz`	Russell Standish, Canberra
`ccc032u@aucc4341.aukuni.ac.nz`	R. Fulton, Auck.

In the USA:

`c3ar@zaphod.uchicago.edu`	Walter Carlip, Chicago
`tnieland@aamrl.af.mil`	Ted Nieland, Dayton
`spencer@cis.ohio-state.edu`	S. Spencer, Columbus

In the UK and Europe:

`abbottp@aston.ac.uk`	Peter Abbott, UK
`texline@vaxa.cc.ic.ac.uk`	Malcolm Clark, UK
`nikunen@cc.helsinki.fi`	Martti Nikunen, Helsinki

I'd like to hear from people interested in distributing OzTEX in other countries. Here's how to get in touch:

Andrew Trevorrow
Kathleen Lumley College
North Adelaide, SA, 5006, Australia
Telephone: (08) 267 1060
Email: `atrevorrow@g.ua.oz` (ACSnet)

Tutorials

`\string` and `\csname`

Stephan v. Bechtolsheim

This article discusses `\string` and `\csname` to convert back and forth between strings and tokens. To control loading macro source files in a convenient way, I will show an application of `\csname`. I will also discuss cross referencing which relies on `\csname`.

Converting Tokens to Strings, `\string`

"`\string <token>`" causes TEX to read the token <token> following `\string` without expansion. Subsequently <token> is replaced by a string representing it. Let me start with some examples.

1. `{\tt\string\hskip}` prints `\hskip`.
2. `{\tt\string$}` prints `$`.
3. `{\tt\string\$}` prints `\$`.
4. `{\tt\string{}` prints `{`.
5. `{\tt\string}}` prints `}`.

Also note:

1. The escape character printed in the previous examples is the backslash. Any other character could be printed by assigning a different character code to `\escapechar`. The default is obviously `\escapechar = '\\`, which assigns the character code of the backslash. If you change `\escapechar` to a negative value, then no escape character is printed:
`\escapechar = -1 \string\xx` prints `xx`.
2. There is an important difference between '`xx`' entered as an ordinary string and '`xx`' generated using `\string` as just shown. All characters generated by `\string` have the category code 12 ("other"), whereas '`x`' ordinarily has category code 11 ("letter").
3. Observe the use of the typewriter font (`\tt`). If you use the roman font and simply write `\string\hskip` the output reads "hskip and *not* \hskip, as expected. The reason for this is that the roman font contains an opening double quote in the position where the typewriter font contains a backslash.
4. `\string` converts only the token following it into a string. For instance, to print two consecutive `$$` you have to repeat `\string` and enter `{\tt\string$\string$}`. If you enter only `{\tt\string$$}` the first dollar sign is