TEXLive CD-ROM

An introduction to TEX Live 4

Editor: Sebastian Rahtz
sebastian.rahtz@oucs.ox.ac.uk

1 Introduction

This article[1] is rewritten from that published in *TUGboat* for TEX Live 3 in 1998 and describes the main features of the TEX Live 4 CD-ROM—a TEX/LATEX distribution for Unix, Linux, Windows32 (and other) systems, and a wide-ranging set of macros, fonts and documentation conforming to the *TEX Directory Standard* (TDS)—which can be used with nearly every TEX setup.

The CD-ROM bundled with this issue of *TUGboat* is being provided as a benefit of 1999 TUG membership. To keep up-to-date on the the TEX Live project, please visit its Web page.[2] The TEX Users Group recognizes the importance of the TEX Live CD-ROM and supports its development and production. Volunteers to assist with this work are encouraged to contact `tex-live@tug.org`.

A fuller version of this document (in English, French, German and Slovak) can be found on the CD-ROM in `tldoc`.

1.1 Changes since TEX Live 3

Although there have been no structural changes, a very great many changes have been made, some more visible than others. Changes that users should know about include:

1. The main TEX programs are based on Web2c version 7.3;
2. Both Unix and Win32 versions are identical to teTEX 0.9 (as of the end of March 1999), and simply add more programs, and a much larger support tree;
3. New programs include `dvipdfm` (DVI to PDF driver) and `tth` (TEX to HTML converter), as well as new versions of pdfTEX, Ω, and ε-TEX;
4. The 'December 1998' (actually March 1999) LATEX is included;
5. A brand new Windows install program is provided;
6. A great many font and macro packages have been updated;
7. Packages are now starting to be classified as 'free' or 'non-free' (according to the Debian Free Software Guidelines[3]) and we expect during the coming year to complete this work, and be able to offer a genuinely 'free' TEX CD-ROM at the start of 2000.

Naturally, much effort has been expended on testing the structural integrity of the `texmf` tree (in particular, checking it against what teTEX does, and checking the licensing conditions of packages).

We very strongly urge any package authors reading this to consider looking at how their work

[1] The guide to `kpathsea` and `web2c` was written by Michel Goossens, and Fabrice Popineau wrote the section on Windows installation and use. The TEX Live CD-ROM distribution is a joint effort by the TEX Users Group, and the UK, French, German, Czech/Slovak, Dutch, Indian and Polish user groups. For the 1999 edition, we are particularly grateful to:

- Karl Berry, who provided the original Web2c distribution, and has continued to give invaluable advice, encouragement, and help;
- Mimi Burbank, who arranged access at the Florida State University Supercomputer Research Institute to a slew of different computers to compile TEX on, and acted as an essential guinea-pig whenever asked;
- Kaja Christiansen, who provided essential feedback and documentation preparation;
- Thomas Esser, without whose marvelous teTEX package this CD-ROM would certainly not exist, and whose continual help makes it a better product;
- Eitan Gurari, whose TEX4ht was used to create the HTML version of this documentation, and who worked tirelessly to improve it at short notice;
- Art Ogawa and Pat Monohon, who coordinated this release for TUG;
- Petr Olšák, who coordinated and checked all the Czech/Slovak material very carefully;
- Fabrice Popineau, who has has worked away unceasingly at the Win32 part of the package (especially the setup!) and contributed in many different ways with ideas, advice and code;
- Walter Schmidt, who checked the emTEX and OS/2 material, and found other horrors;
- Staszek Wawrykiewicz, who provided great checking feedback, and coordinated the Polish contributions;
- Olaf Weber, for his patient assembly and maintenance of Web2c 7.3;
- Graham Williams, on whose work the catalogue of packages depends.

Alain Rabaute, Pascal Quignon, Gerhard Wilhelms, Fabrice Popineau, Libor Skarvada, Staszek Wawrykiewicz, Erik Frambach, and Ulrik Vieth kindly translated documentation into their respective languages, checked other documentation, and provided very welcome feedback.

[2] `http://www.tug.org/tex-live.html`

[3] `http://www.debian.org/intro/free`

is arranged on the CD-ROM, and contacting us with any problem. But most importantly, look at the *Catalogue* maintained by Graham Williams[4] and check your details in there, especially the licensing! Future versions of TeX Live will rely more and more on the *Catalogue*.

2 Structure and contents of the CD-ROM

The important CD-ROM top-level directories are listed below.

bin The TeX family programs, arranged in separate platform directories;

tldoc Documentation for TeX Live;

FAQ Frequently Asked Questions, in English, French, and German;

info Documentation in GNU 'info' format for the TeX system;

man Documentation in the form of Unix man pages for the TeX system;

source The source of all programs, including the main Web2c, TeX, and METAFONT distributions – stored in a compressed tar archive;

support Various bits of TeX-related software which are *not* installed by default, such as MusixTeX, support programs, and a complete distribution of Ghostscript version 5.50;

systems Packaged TeX systems which are separate from the main TeX Live. Subdirectories in here are:

- **macintosh** The CMacTeX package ready to install;
- **msdos** The emTeX package for MS-Dos;
- **os2** The OS/2 TeX package emTeX/TDS and the EPMTFE TeX shell for the EPM editor.

texmf The main support tree of macros, fonts and documentation;

usergrps Material about TeX User Groups.

2.1 The TDS tree

The TeX Live `texmf` tree consists of various 'collections', each of which has a set of 'packages', of which there are over 400 on the CD-ROM. Normal installation allows the user to copy all of a collection to a local hard disk from the CD-ROM, but it is also possible to install just one package of a collection. The collections are:

ams The American Mathematical Society macro packages and fonts.

bibtex BibTeX styles and databases.

doc General guides and documentation in various formats, including HTML and PDF.

dvips Support for Rokicki's DVI-to-PostScript driver.

etex Support for ε-TeX.

fonts Font sources, metrics, PostScript and bitmap forms.

formats Eplain, RevTeX, phyzzx, texsis, alatex, text1, lollipop, etc.

generic Extra macros for use with any format.

graphics Macro packages for graphics.

lang Support for non-English languages.

latex LaTeX, including official tools and all LaTeX 2_ε contributed packages.

metapost Support for MetaPost.

omega Support for Ω.

pdftex Support for pdfTeX

plain Macros for plain TeX.

systems Binaries for Unix and Win32 platforms.

texlive Basic material for the distribution.

Each of the collections is divided into *basic* (1), *recommended* (2), and *other* (3). Thus all packages in collection `latex1` are what one must have to get started with LaTeX, packages in `latex2` are recommended for most users, and `latex3` contains optional packages. The directory `texmf/lists` contains lists of all files in each package (used by the installation programs). Graham William's *Catalogue* lists all the packages, noting whether they are in TeXLive, and giving details.

3 Installation and use under Unix

You can use the TeX Live CD-ROM in three ways:

1. You can mount the CD-ROM on your file system, adjust your `PATH`, and run everything off the CD-ROM; this takes very little disk space, and gives you immediate access to everything on the CD-ROM; although the performance will not be optimal, it is perfectly acceptable on, for instance, PCs running Linux.
2. You can install all or part of the system to your local hard disk; this is the best method for many people, if they have enough disk space to spare (a minimum of about 10 megabytes, or 100 megabytes for a recommended good-sized system).
3. You can install selected packages to work either with your existing TeX system or a TeX Live system you installed earlier.

Each of these methods is described in more detail in the following sections.

[4] `http://www.cmis.csiro.au/Graham.Williams/TeX/catalogue.html`

Warning: This CD-ROM is in ISO 9660 (High Sierra) format, with Rock Ridge and Joliet extensions. In order to take full advantage of the CD-ROM on a Unix system, your system needs to be able to use the Rock Ridge extensions. Please consult the documentation for your mount command to see if it is possible. If you have several different machines on a local network, see if you can mount the CD-ROM on one which *does* support Rock Ridge, and use this with the others.

3.1 Running from the CD-ROM under Unix

The organisation of Web2c means that you can run programs simply by adding the appropriate directory under `bin` on the CD-ROM to your `PATH`, and the support files will all be found with no further ado. The following shows the list of available systems and the corresponding directories.

DEC Alpha OSF/1	`alpha-osf4.0`
HP9000 HPUX	`hppa11-hpux10.10`
Intel Linux	`i386-linux`
	`i386-linux-libc5`
SGI IRIX	`mips-irix6.2`
IBM RS 6000 AIX	`rs6000-aix4.1.4`
Sun Sparc Solaris	`sparc-solaris2.5.1`
Windows 95 or NT (Intel)	`win32`

You may worry that when you subsequently make fonts or change configuration, things will go wrong because you cannot change files on the CD-ROM. However, you can maintain a parallel, writeable, TEX tree on your hard disk; this is searched before the main tree on the CD-ROM. The default location is `texmf-localconfig` on the CD-ROM (which does not exist!), so you *must* override this by setting the `VARTEXMF` environment variable.

Thus *sh* or *bash* users on an Intel PC running Linux can mount the TEX Live CD-ROM on `/cdrom` by issuing the command:

```
>> mount -t iso9660 /dev/cdrom /cdrom
```

Then they should include the directory containing the binaries for the given architecture into the search path by updating the `PATH` variable.

```
PATH=/cdrom/bin/i386-linux:$PATH
export PATH
VARTEXMF=/usr/TeX.local
export VARTEXMF
```

For convenience, these statements can also be entered into the `.profile` script.

If in doubt, ask your local system support guru to help you work out how to mount your CD-ROM or which directory to use for your system.

Appropriate support files will be installed on your hard disk the first time you need them. It is a good idea to immediately run the `texconfig` script to initialise things, and check it all works.

3.2 Installing to a hard disk

All of the necessary steps to install all or part of the distribution on your hard disk are achieved by mounting the CD-ROM, changing to the top-level directory, and typing:

```
>> sh install-cd.sh
```

(On some Unix systems, you may need to use sh5 or bash.) This script works by accessing lists of collections and packages from the CD-ROM, and trying to guess what sort of computer system you are on. It should start by displaying the following:

```
Initializing collections... Done initializing.
Counting selected collections... Done counting.
Calculating disk space requirements for collections...
         Done calculating that.
Initializing system packages...
         Done initializing system.
```

It will then show the main control screen (Figure 1), which lets you change four things:

1. the type of system you are on, or want to install for;
2. the collections you want to install, at the *basic*, *recommended* or *other* level;
3. the location on your hard disk to put the files;
4. some runtime behaviour features.

You choose options by typing a letter or number and pressing 'return'. In the example, a Linux ELF system has been detected, the default of all collections to *recommended* level has been chosen, and the default installation directory is `/usr/TeX`; note that the disk space required for the current installation configuration is also displayed. If you make a suggested setup, you need about 100 megabytes of disk free; however, the basic setup will only take about 10 megabytes, and you can enhance it with selected packages as you need them.

Under the directory you choose for installation, the installation script will put the binaries in a subdirectory of `bin`, and the support tree in `texmf`.

The `options` item lets you decide whether to make new fonts be created in another location (if you want the main package mounted read-only for most users), and whether to make symbolic links for the man and GNU info pages in the 'standard' locations; you'll need 'root' permissions for tasks to do this, of course.

When you choose `<C>` for 'collections', you will see the display of available collections, the level of

```
==================> TeX Live installation procedure <==================

===> Note: Letters/digits in <angle brackets> indicate menu items <===
===>       for commands or configurable options                    <===

    Proposed platform: Intel x86 with GNU/Linux
    <P> over-ride system detection and choose platform
    <C> collections:    24 out of 34, disk space required: 9812099 kB
    <S> systems:         1 out of  8, disk space required:   7925 kB
                                 total disk space required: 9820024 kB
    <L> install level (1: basic, 2: recommended, 3: all): 2
    <D> directories:
      TEXDIR     (The main TeX directory)            : /usr/TeX
      TEXMFLOCAL (TeX directory for local styles etc): /var/TeX-local
    <O> options:
       [ ] alternate directory for generated fonts ()
       [ ] alternate directory for configuration ()
       [ ] create symlinks in standard directories
       [ ] do not install macro/font doc tree
       [ ] do not install macro/font source tree
       [ ] only install free software
    <I> start installation,  <H> help,  <Q> quit
Enter command:
```

Figure 1: Main control screen

```
       name            selection         size
  <1>  bibtex     [recommended]     7597 kB
  <2>  doc        [recommended]    21152 kB
  <3>  dvips      [recommended]      430 kB
  <4>  etex       [recommended]      102 kB
  <5>  fonts      [recommended]    51447 kB
  <6>  formats    [recommended]    14651 kB
  <7>  generic    [recommended]      459 kB
  <8>  graphics   [recommended]     9674 kB
  <9>  lang       [recommended]    19618 kB
  <U>  latex      [recommended]    23429 kB
  <V>  metapost   [recommended]     1443 kB
  <W>  omega      [recommended]     4986 kB
  <X>  pdftex     [recommended]      471 kB
  <Y>  plain      [recommended]     1113 kB
  <Z>  texlive    [recommended]    10155 kB
                            SUM:  166829 kB
================================================================================
  global commands: select <N>one / <B>asic / R<E>commended / <A>ll
                   for all collections
  <R>  return to platform menu
  <Q>  quit
```

Figure 2: Selecting collections

```
Collection: Fonts
==================================================

Fonts, including metrics, virtual fonts and sources

==================================================
  <N>  No packages
  <B>  Basic packages                  [  1023 kB]
  <E>  Basic + Recommended packages    [ 51447 kB]
  <A>  All packages                    [127417 kB]

==================================================
  <R>   return to collection menu
  <Q>   quit

Enter command:
```

Figure 3: Customizing a collection

installation selected, and the disk space required (Figure 2). You can set alternative levels of installation for each collection, ranging from *none* to *all*. You can either set this for all collections at once, or choose a particular collection and set its level (Figure 3).

When you are finished, return to the main screen, and ask the installation to start. It will take each of the collections and systems that you requested, consult the list of files on the CD-ROM, and build a master list of files to transfer. These will then be copied to your hard disk. If you installed a system, an initialisation sequence is now run (creating format files, etc.). When this has finished, all you need do is add the correct subdirectory of `bin` in the TEX installation to your path, and start using TEX. If you want, you can move the binaries up one level, e.g., from `/usr/local/bin/alpha-osf3.2` to `/usr/local/bin`; if you do this, however, you must edit `texmf/web2c/texmf.cnf` (see Appendix 7) and change the line near the start which reads

```
TEXMFMAIN = $SELFAUTOPARENT
```

to

```
TEXMFMAIN = $SELFAUTODIR
```

If you move the whole installation to another directory tree entirely, you need to edit `TEXMFMAIN` to specify the support tree explicitly, and set `TEXMFCNF` in your environment to `$TEXMFMAIN/texmf/web2c`.

3.3 Installing individual packages to a hard disk

You may want to use the TEX Live CD-ROM to either update an existing setup, or add features to an earlier installation from the CD-ROM. The main installation program is intended for the first time only, and subsequently you should use the `install-pkg.sh` script on the CD-ROM. Run this by mounting the CD-ROM, changing to the mounted directory, and typing

```
>> sh install-pkg.sh options
```

The script supports nine options; the first four let you set the individual package you want to install, the whole collection (i.e., `ams2`), the name of the mounted CD-ROM directory, and the name of the directory containing the list files (normally these latter two will be set automatically):

`--package=`*name*
`--collection=`*name*
`--cddir=`*name*
`--listdir=`*name*

What actually happens is controlled by four more switches; the first two allow you to exclude documentation or source files from the installation, the third stops the default action of running `mktexlsr` on completion to rebuild the file database, and the last does nothing but list the files that would be installed:

`--nodoc`
`--nosrc`
`--nohash`
`--listonly`

Finally, you can specify that, instead of installing the files, the script should make a tar archive in a specified location:

`--archive=`*name*

Thus, if we simply wanted to see the files that make up the package fancyhdr before we installed it, our command and output would be as follows:

```
>> sh install-pkg.sh --package=fancyhdr \
>>                           --listonly
texmf/doc/latex/fancyhdr/fancyhdr.dvi
texmf/doc/latex/fancyhdr/fancyhdr.tex
texmf/lists/free/latex3/fancyhdr
texmf/source/latex/fancyhdr/README
texmf/tex/latex/fancyhdr/extramarks.sty
texmf/tex/latex/fancyhdr/fancyhdr.sty
texmf/tex/latex/fancyhdr/fixmarks.sty
```

Other examples of usage are:

• Install the LaTeX package `natbib`:

```
>> sh install-pkg.sh --package=natbib
```

• Install the LaTeX package `alg` with no source files and no documentation:

```
>> sh install-pkg.sh  --package=alg \
>>                  --nosrc --nodoc
```

• Install all the packages available in the *other* Plain TeX collection:

```
>> sh install-pkg.sh --collection=plain3
```

• Place all files which are needed for PSTricks in a tar file in `/tmp`:

```
>> sh install-pkg.sh --package=pstricks \
>>          --archive=/tmp/pstricks.tar
```

3.4 The texconfig program

After the installation program has copied all files to their final locations, you can use a program called texconfig that allows you to configure the system to fit your local needs. This can be called at any other time to change your setup, with a full-screen (which requires the dialog program, included as part of the binary packages) or command-line interface. It should be used for all maintenance, such as changes of installed printers, or rebuilding the file database. Both modes have help text to guide you through the facilities.

4 Installation and use under Windows

This section only applies to systems running Windows 9x or NT. If you run Windows 3.1, you will have to install `emtex` from the top level `systems` directory by hand.

It is also necessary to have your Windows set up so that it uses the Microsoft Joliet extensions for reading CD-ROMs; simply look at the CD-ROM in Windows Explorer and see whether it shows long, mixed-case file names. If it does not, you cannot use the ready-to-run system on the CD-ROM.

4.1 What is fpTeX?

The system on the CD-ROM for Windows is Fabrice Popineau's fpTeX. This is a port to Windows 9x and Windows NT– referred to as Win32 – of the well known distribution teTeX for Unix. More precisely, given obvious differences between Unix and Win32, some things behave differently under fpTeX, some are still missing, some are just different, but the large majority behave just the same as under Unix.

4.2 What is in this port

This version includes all the same programs as the Unix side of TeX Live, with some additions which can be installed.

Among the programs, there are a few DLLs:

- the ones that begin with msvc are the Microsoft C library targeted for multi-threaded applications,
- the Kpathsea dynamic-linked library,
- `zlib.dll` (compress library) and `libpng.dll` (Portable Network Graphics) for pdfTeX and a few other programs,
- `tex.dll`, `pdftex.dll` and a few others refer to TeX engines – see the explanation below.

All the various TeX engines are distributed in the form of a `.dll` file for the core, and a `.exe` file for the front-end. This is the answer to the problem of link files that do not exist on Win32 platforms. For example, the TeX engine is made up of:

```
11/19/98  11:07a      217,088 tex.dll
11/19/98  11:07a       16,384 tex.exe
```

and the `latex.exe` file is nothing but a rough copy of `tex.exe` using the same core `tex.dll`. The same approach is used for the `mktex*.exe` family of programs which are linked to the `mktex.dll` library.

4.3 Running from the CD-ROM

You can run all the TeX programs directly off the CD-ROM, and have access to all the macros and fonts immediately, at the price of a slower performance than if you install on the hard disk. To do this, you must add the `bin/win32` directory of the CD-ROM to your `PATH`, using the Windows configuration software. Now you can run the programs at a command prompt, or use the shareware WinEdt editor, which runs the programs from convenient menus.

4.4 How to install it

When you put TeX Live in your computer, the setup should run automatically; if it does not, run the program `autorun.exe`. Now follow the instructions—here are some hints:

- Choose a *root* for your installation, `c:\TeX` is proposed by default, but you can change it because you will need lots of disk space: more than 300Mb for a full installation, and beware that the cluster size on FAT partitions will make the package appear even bigger;
- Do not use any path with embedded 'space' character: TEX won't like that. The `setup.exe` checks for that anyway;
- Your *main* `texmf` tree will be *root*`/texmf` and designated by the variable `$TEXMFMAIN`;
- You have the opportunity to add some more `texmf` trees:
  - one *local* `texmf` tree, which is designated by the variable `$TEXMFLOCAL` and is by default *root*`/texmf.local`. It is intended to store your site local macros and style files, and also any locally generated font files. *If you do not specify a local `texmf` tree, you will be asked to set the variable `VARTEXFONTS` to point to some directory where those fonts will be stored. If you specify a local `texmf` tree, it will be used for those fonts.*
  - one *home* `texmf` tree, which is designated by the variable `$HOMETEXMF` and is by default `$HOME/texmf`. This is meaningful only under Windows NT, where users have a `$HOME`. Usually, Windows 9x users do not have a `$HOME`, so they should leave this place empty;

  These locations can be edited manually by looking for their variables names in the file `texmf/web2c/texmf.cnf`.
- You will be asked if you want to install:
  - PK fonts,
  - only *free* packages, in which case some packages will even disappear from the list for a *custom* installation,
  - if you want the documentation accompanying each of the packages being installed; guides and general documentation will always be installed, but not the material that is provided with some specific package;
  - *idem* for source files.
- Choose your setup from *basic*, *recommended* and *full*. If you already know what you want to install, you can also choose to do a *custom* setup. In this case, you will be presented with a list of collections of packages. Each collection is (de)selectable by itself, and so is each package individually. A short description is provided for many packages, thanks to the Web catalogue of Graham Williams.
- Ignore the 'setup' collection in the list of collections. It refers to the files in the `\setupw32` on the CD-ROM and should have been made invisible;
- You will have the opportunity to add some more special packages, namely Ghostscript and Ghostview, the Postscript interpreter and previewer, ImageMagick which allows for image manipulations and conversions, WinEdt which is a good shareware environment for editing and typesetting with TEX, and texshell which does quite the same as WinEdt (free but less fancy).
- You can review your installation settings, and if everything is okay, the file transfer will begin;
- Once it is done, the installers for Ghostview and WinEdt will be called if selected. The installation and configuration of everything else is handled by `setup.exe`. It is completely automatic. The last step is the build of 'ls-R' files;
- Eventually, the setup will be over and the documentation displayed using your default Web browser. Windows 9x users will need to reboot before being able to run anything.

A number of items will appear under the `Start->Programs->TeXLive` menu.

4.5 What does the setup hide from me ?

If you want to hack the configuration by yourself, here is a more detailed description of what the `setup` does — and what it does not. Your `PATH` is modified to make the programs installed accessible. It is checked for any older version of fpTEX or TEX-Live, and if one is found, its entry in `PATH` is removed. This is done by looking for the file `kpathsea.dll` along your `PATH`.

If Ghostscript installation has been requested, the location of `gs5.50` is added to your `PATH` because the files `gswin32c.exe` (command line interface to Ghostscript) and `gsdll32.dll` (dll embedded Ghostscript) are accessed by several programs of the distribution. Ghostscript uses the registry now (version 5.50) and so does not need any other setting to find the relevant files. Previous or customized installations might require to set the `GS_LIB` variable. See the appropriate documentation.

ImageMagick is also added to your `PATH` if it has been selected. Moreover, the right *delegates.mk* file is copied according to your platform (Windows 9x or Windows NT). See *ImageMagick* documentation for more details.

Your main <root>/texmf/web2c/texmf.cnf file is edited to reflect the texmf trees you have specified, and the location of locally generated fonts. The variables $TEXMF, $TEXMFLOCAL, $HOMETEXMF, and $VARTEXFONTS are modified.

The file <root>/texmf/web2c/mktex.cnf is edited to add the feature 'varfonts', which will force any locally generated font to be stored in $VARTEXFONTS.

The configuration for tex4ht is undertaken by editing <root>/texmf/tex4ht/base. The only relevant part is that it needs to run the convert.exe tool of *ImageMagick*, so the full path to access it is provided.

4.6 Testing the installation

A valuable tool to test the installation now is the program kpsewhich.

As a first step, you should check if Web2C correctly identifies the location of your texmf tree. Open a command prompt window and type

```
kpsewhich -expand-path=$TEXMF
```

The answer should be the location of your texmf tree (e.g., c:/TeX/texmf if you unpacked the archive files as in the example above —note that the answer is a Unix style path, i.e., the MS-Dos style \\ is substituted by /; you don't have to worry about this). If you do not get the right answer you have probably changed the default directory structure. In this case you have to set the variable TEXMF manually to the root directory of your texmf tree.

If you want to be on the safe side, you may type in mktexlsr to update the ls-R database, even though a proper ls-R file should be provided after installation.

4.7 Network installation and filesystem considerations

All the support files, everything except the files in bin/win32, are shareable with a Unix installation. This means you can use Samba either to mount from a Windows NT server to a Unix workstation or the converse. Several strategies are possible:

- Put everything on the server. Just add each set of files for the operating system and architecture you want to use in the bin directory. That means for example bin/win32 and bin/i386-linux.
- Install a local copy of the binaries and format files. In this case, assign $TEXMFMAIN to the main texmf tree that will lie on the network.

These schemes should have been handled by the InstallShield installer. But so many problems arose with this installer that these features have been delayed to the next version of the setup program.

Win32 supports multiple filesystems:

- MS-Dos FAT, 8.3 and uppercase filenames
- Protected mode FAT, long filenames, but case-insensitive
- NTFS, long filenames and case-sensitive
- ISO9660 CD-ROM, 8.3 and uppercase filenames

Moreover, Win32 calls which refer to filenames are case-insensitive, and there are several other features in NTFS that Win32 can't use for the moment. Another dimension is the use of different directory separators: / or \, but Win32 calls accept both.

So what difficulties may arise ?

Most likely, you will have some style files with long filenames. If you are running on a filesystem which supports them,[5] there is no problem and you have nothing to do. Otherwise, you will need to use the alias feature of Kpathsea. Suppose, for instance, you are trying to install texmf on a FAT partition and you have the style file named longtable.sty in your tree. The filename will be truncated to its 8.3 form: longtabl.sty. In this case, you will need to create a file named aliases in the same place as the ls-R file in your texmf tree. This file should contain the following line:

```
longtabl.sty longtable.sty
```

All references to longtable.sty will be redirected to longtabl.sty if the long filename is not found.

Otherwise, if you think you have trouble with filenames, consider doing the following:

- paths in config files and environment variables should be written using / rather than \;
- ls-R databases should be in lower case, even if you are running on FAT or CD-ROM;
- use the debug feature of Kpathsea and kpsewhich to demonstrate your problem and email us the results of your investigations.

5 Building on a new Unix platform

If you have a platform for which we have not provided binary sources, you will need to compile TEX and friends from scratch. This is not as hard as it sounds. What you need is all in the directory source on the CD-ROM.

You should first install the support tree from the TEX Live CD-ROM (do a basic install, with no system binaries chosen).

[5] For example, NTFS but not FAT!

5.1 Prerequisites

You will need about 100 megabytes of disk space to compile all of TEX and its support programs. You'll also need an ANSI C compiler, a make utility, a lexical scanner, and a parser generator. The GNU utilities (gcc, GNU make, m4, flex, bison) are the most widely tested on different platforms. gcc-2.7.* flex-2.4.7 and GNU make-3.72.1 or newer should work well. You may be able to work with other C compilers and make programs, but you will need a good understanding of building Unix programs to sort out problems. The command `uname` must return a sensible value.

5.2 Configuration

First, unpack the source from the compressed `tar` file in the directory `source` to your disk and change directory to where you placed it. Decide where the 'root' of the installation will be, e.g., `/var/TeX` or `/usr/TeX`. Obviously you should use the same location that you specified when you installed the support tree.

Now, start the build process by running configure with a command-line like

```
>> ./configure -prefix=/usr/TeX
```

The 'prefix' directory is the one where you installed the support tree; the directory layout that will be used is as follows (where $TEXDIR stands for the directory you chose):

`$TEXDIR/man`	Unix manual pages
`$TEXDIR/share/texmf`	main tree with fonts, macros, etc
`$TEXDIR/info`	GNU 'info' manuals
`$TEXDIR/bin/$PLATFORM`	binaries

You can omit the use of '`share/`' part for the `texmf` directory if you want, as `$TEXDIR/share/texmf` and `$TEXDIR/texmf` are auto-detected by configure. If you choose something different, you have to specify that directory with the `--datadir` option of configure.

If you want to leave out the `$PLATFORM` directory level (that is, put binaries directly into `$TEXDIR/bin`), you can use the configure option `--disable-multiplatform`.

Have a look at the output of `./configure --help` for more options you can use (such as omitting optional packages such as Ω or ε-TEX).

5.3 Running make

Make sure the shell variable `noclobber` is not set, and then type

```
>> make world
```

and relax....

It could also be useful to log all the output, e.g., by typing

```
>> sh -c "make world >world.log 2>&1" &
```

Before you think that everything is okay, please check the log file for errors (GNU make always uses the string "`Error:`" whenever a command returns an error code) and check if all binaries are built:

```
>> cd /usr/TeX/bin/i586-pc-linux-gnu
>> ls | wc
```

The result should be 204. `make world` is equivalent to `make all install strip`

If you need special privileges for `make install`, you can run two make jobs in separate runs:

```
>> make all
>> su
>> make install strip
```

5.4 Final configuration steps

Set up your `PATH` to include the directory containing the just-installed binaries (e.g., `/usr/TeX/bin/mips-sgi-irix6.3`); similarly, `MANPATH` and `INFOPATH` to include the relevant newly installed subdirectories, i.e., `$TEXDIR/man` and `$TEXDIR/info`.

The program texconfig allows you to set the defaults for hyphenation, paper size, print command, METAFONT mode, etc. You can run this command interactively and see what options it offers, or type

```
>> texconfig help
```

For example, if you are not using A4 format paper, you can make 'lettersize' the default using:

```
>> texconfig dvips paper letter
>> texconfig xdvi paper us
```

6 A user's guide to the Web2c system

Web2c contains a set of TEX-related programs, i.e., TEX itself, METAFONT, MetaPost, BIBTEX, etc; it works on Unix, Windows 3.1, 9x/NT, DOS, and other operating systems. It uses Knuth's original sources for TEX and other basic programs written in `web` and translates them into C source code. Moreover, the system offers a large set of macros and functions developed to augment the original TEX software. The core TEX family components are:

bibtex Maintaining bibliographies;

dmp troff to MPX (MetaPost pictures);

dvicopy Produces modified copy of DVI file;

dvitomp DVI to MPX (MetaPost pictures);

dvitype DVI to human-readable text;
gftodvi Generic font proofsheets;
gftopk Generic to packed fonts;
gftype GF to human-readable text;
makempx MetaPost label typesetting;
mf Creating typeface families;
mft Prettyprinting METAFONT source;
mpost Creating technical diagrams;
mpto MetaPost label extraction;
newer Compare modification times;
patgen Creating hyphenation patterns;
pktogf Packed to generic fonts;
pktype PK to human-readable text;
pltotf Property list to TFM;
pooltype Display `web` pool files;
tangle `web` to Pascal;
tex Typesetting;
tftopl TFM to property list;
vftovp Virtual font to virtual property list;
vptovf Virtual property list to virtual font;
weave `web` to TEX.

The precise functions and syntax of these programs are described in the documentation of the individual packages or of Web2c itself. However, knowing a few principles governing the whole family of programs will help you to benefit optimally from your Web2c installation.

All programs honor the standard GNU options:

`--help` print basic usage summary.
`--verbose` print detailed progress report.
`--version` print version information, then exit.

For locating files the Web2c programs use the path searching library Kpathsea. This library uses a combination of environment variables and a few configuration files to optimize searching the TEX directory tree. Web2c 7.3 can handle more than one directory tree simultaneously, which is useful if one wants to maintain TEX's standard distribution and local extensions in two distinct trees. To speed up file searches the root of each tree has a file, `ls-R`, containing an entry showing the name and relative pathname for all files "hanging" under that root.

6.1 Kpathsea path searching

Let us first describe the generic path searching mechanism of the Kpathsea library.

We call a *search path* a colon- or semicolon-separated list of *path elements*, which are basically directory names. A search path can come from (a combination of) many sources. To look up a file "`my-file`" along a path "`.:/dir`", Kpathsea checks each element of the path in turn: first `./my-file`, then `/dir/my-file`, returning the first match (or possibly all matches).

In order to adapt optimally to all operating systems' conventions, on non-Unix systems Kpathsea can use filename separators different from "colon" ("`:`") and "slash" ("`/`").

To check a particular path element p, Kpathsea first checks if a prebuilt database (see "Filename database" on page 31) applies to p, i.e., if the database is in a directory that is a prefix of p. If so, the path specification is matched against the contents of the database.

If the database does not exist, or does not apply to this path element, or contains no matches, the filesystem is searched (if this was not forbidden by a specification starting with "`!!`" and if the file being searched for must exist). Kpathsea constructs the list of directories that correspond to this path element, and then checks in each for the file being sought.

The "file must exist" condition comes into play with "`.vf`" files and input files read by TEX's `\openin` command. Such files may not exist (e.g., `cmr10.vf`), and so it would be wrong to search the disk for them. Therefore, if you fail to update `ls-R` when you install a new "`.vf`" file, it will never be found. Each path element is checked in turn: first the database, then the disk. If a match is found, the search stops and the result is returned.

Although the simplest and most common path element is a directory name, Kpathsea supports additional features in search paths: layered default values, environment variable names, config file values, users' home directories, and recursive subdirectory searching. Thus, we say that Kpathsea *expands* a path element, meaning it transforms all the specifications into basic directory name or names. This is described in the following sections in the same order as it takes place.

Note that if the filename being searched for is absolute or explicitly relative, i.e., starts with "`/`" or "`./`" or "`../`", Kpathsea simply checks if that file exists.

6.1.1 Path sources

A search path can come from many sources. In the order in which Kpathsea uses them:

1. A user-set environment variable, for instance, `TEXINPUTS`. Environment variables with a period and a program name appended override; e.g., if "`latex`" is the name of the program

```
TEXMF              = {$TEXMFLOCAL;!!$TEXMFMAIN}
TEXINPUTS.latex    = .;$TEXMF/tex/{latex;generic;}//
TEXINPUTS.fontinst = .;$TEXMF/tex//;$TEXMF/fonts/afm//
% e-TeX related files
TEXINPUTS.elatex   = .;$TEXMF/{etex;tex}/{latex;generic;}//
TEXINPUTS.etex     = .;$TEXMF/{etex;tex}/{eplain;plain;generic;}//
```

Figure 4: An illustrative configuration file sample

being run, then `TEXINPUTS.latex` will override `TEXINPUTS`.

2. A program-specific configuration file, for example, a line "`S /a:/b`" in dvips's `config.ps`.
3. A Kpathsea configuration file `texmf.cnf`, containing a line like "`TEXINPUTS=/c:/d`" (see below).
4. The compile-time default.

You can see each of these values for a given search path by using the debugging options (see "Debugging actions" on page 33).

6.1.2 Config files

Kpathsea reads *runtime configuration files* named `texmf.cnf` for search path and other definitions. The search path used to look for these files is named `TEXMFCNF` (by default such a file lives in the `texmf/web2c` subdirectory). *All* `texmf.cnf` files in the search path will be read and definitions in earlier files override those in later files. Thus, with a search path of `.:$TEXMF`, values from `./texmf.cnf` override those from `$TEXMF/texmf.cnf`.

While reading the description of the format of the file `texmf.cnf` below, please also refer to appendix 7, starting on page 37, which lists the `texmf.cnf` file on the CD-ROM.

- Comments start with "`%`" and continue to the end of the line.
- Blank lines are ignored.
- A `\` at the end of a line acts as a continuation character, i.e., the next line is appended. Whitespace at the beginning of continuation lines is not ignored.
- Each remaining line has the form:

  variable`[.`*progname*`] [=]` *value*

  where the "`=`" and surrounding whitespace are optional.
- The "*variable*" name may contain any character other than whitespace, "`=`", or "`.`", but sticking to "`A-Za-z_`" is safest.
- If "`.`*progname*" is present, the definition only applies if the program that is running is named *progname* or *progname*`.exe`. This allows different flavors of TEX to have different search paths, for example.
- "*value*" may contain any characters except "`%`" and "`@`". The "`$`*var*`.`*prog*" feature is not available on the right-hand side; instead, you must use an additional variable. A "`;`" in "*value*" is translated to "`:`" if running under Unix; this is useful to be able to have a single `texmf.cnf` for Unix, MS-Dos and Windows systems.
- All definitions are read before anything is expanded, so variables can be referenced before they are defined.

A configuration file fragment illustrating most of these points is shown in Figure 4.

6.1.3 Path expansion

Kpathsea recognizes certain special characters and constructions in search paths, similar to those available in Unix shells. As a general example, the complex path, `~$USER/{foo,bar}//baz`, expands to all subdirectories under directories `foo` and `bar` in `$USER`'s home directory that contain a directory or file `baz`. These expansions are explained in the sections below.

6.1.4 Default expansion

If the highest-priority search path (see "Path sources" on page 29) contains an *extra colon* (i.e., leading, trailing, or doubled), Kpathsea inserts at that point the next-highest-priority search path that is defined. If that inserted path has an extra colon, the same happens with the next highest. For example, given an environment variable setting

```
>> setenv TEXINPUTS /home/karl:
```

and a `TEXINPUTS` value from `texmf.cnf` of

```
.:\$TEXMF//tex
```

then the final value used for searching will be:

```
/home/karl:.:\$TEXMF//tex
```

Since it would be useless to insert the default value in more than one place, Kpathsea changes only one extra "`:`" and leaves any others in place: it checks first for a leading "`:`", then a trailing "`:`", then a doubled "`:`".

6.1.5 Brace expansion

A useful feature is brace expansion, which means that, for instance, `v{a,b}w` expands to `vaw:vbw`. Nesting is allowed. This can be used to implement multiple TEX hierarchies, by assigning a brace list to `$TEXMF`. For example, in `texmf.cnf`, you find the following definition (on one line!):

```
TEXMF = {$HOMETEXMF,$TEXMFLOCAL,
          !!$VARTEXMF,!!$TEXMFMAIN}
```

Using this you can then write something like

```
TEXINPUTS = .;$TEXMF/tex//
```

which means that, after looking in the current directory, the `$HOMETEXMF/tex`, `$TEXMFLOCAL/tex`, `$VARTEXMF/tex` and `$TEXMFMAIN/tex` trees *only*) will be searched (the last two use using `ls-R` data base files). It is a convenient way for running two parallel TEX structures, one "frozen" (on a CD-ROM, for instance) and the other being continuously updated with new versions as they become available. By using the `$TEXMF` variable in all definitions, one is sure to always search the up-to-date tree first.

6.1.6 Subdirectory expansion

Two or more consecutive slashes in a path element following a directory *d* is replaced by all subdirectories of *d*: first those subdirectories directly under *d*, then the subsubdirectories under those, and so on. At each level, the order in which the directories are searched is *unspecified*.

If you specify any filename components after the "`//`", only subdirectories with matching components are included. For example, "`/a//b`" expands into directories `/a/1/b`, `/a/2/b`, `/a/1/1/b`, and so on, but not `/a/b/c` or `/a/1`.

Multiple "`//`" constructs in a path are possible, but "`//`" at the beginning of a path is ignored.

6.1.7 List of special characters and their meaning: a summary

The following list summarises the meaning of special characters in Kpathsea configuration files.

`:` Separator in path specification; at the beginning or the end of a path it substitutes the default path expansion.

`;` Separator on non-Unix systems (acts like `:`).

`$` Variable expansion.

`~` Represents the user's home directory.

`{...}` Brace expansion, e.g., `a{1,2}b` will become `a1b:a2b`.

`//` Subdirectory expansion (can occur anywhere in a path, except at its begining).

`%` Start of comment.

`\` Continuation character (allows multi-line entries).

`!!` Search *only* database to locate file, *do not* search the disk.

6.2 Filename databases

Kpathsea goes to some lengths to minimize disk accesses for searches. Nevertheless, at installations with enough directories, searching each possible directory for a given file can take an excessively long time (this is especially true if many hundreds of font directories have to be traversed.) Therefore, Kpathsea can use an externally-built "database" file named `ls-R` that maps files to directories, thus avoiding the need to exhaustively search the disk.

A second database file `aliases` allows you to give additional names to the files listed in `ls-R`. This can be helpful to adapt to DOS-like "8.3" filename conventions in source files.

6.2.1 The filename database

As explained above, the name of the main filename database must be `ls-R`. You can put one at the root of each TEX hierarchy in your installation that you wish to be searched (`$TEXMF` by default); most sites have only one hierarchy. Kpathsea looks for `ls-R` files along the `TEXMFDBS` path.

The recommended way to create and maintain "`ls-R`" is to run the `mktexlsr` script included with the distribution. It is invoked by the various "`mktex`"... scripts. In principle, this script just runs the command

```
cd /your/texmf/root && ls -LAR ./ >ls-R
```

presuming your system's `ls` produces the right output format (GNU's `ls` is all right). To ensure that the database is always up-to-date, it is easiest to rebuild it regularly via `cron`, so that for changes in the installed files—perhaps after installing or updating a LATEX package—the file `ls-R` is automatically updated.

If a file is not found in the database, by default Kpathsea goes ahead and searches the disk. If a particular path element begins with "`!!`", however, *only* the database will be searched for that element, never the disk.

6.2.2 kpsewhich: Standalone path searching

The `kpsewhich` program exercises path searching independent of any particular application. This can be useful as a sort of `find` program to locate files in TEX hierarchies (this is used heavily in the distributed "`mktex`"... scripts).

```
>> kpsewhich option... filename...
```

The options specified in "*option*" can start with either "-" or "--", and any unambiguous abbreviation is accepted.

Kpathsea looks up each non-option argument on the command line as a filename, and returns the first file found. There is no option to return all the files with a particular name (you can run the Unix "find" utility for that).

The more important options are described next.

--dpi=*num* Set the resolution to "*num*"; this only affects "gf" and "pk" lookups. "-D" is a synonym, for compatibility with dvips. Default is 600.

--format=*name*
Set the format for lookup to "*name*". By default, the format is guessed from the filename. For formats which do not have an associated unambiguous suffix, such as MetaPost support files and dvips configuration files, you have to specify the name as found in the first column of Table 1 on p. 36, which lists currently recognized names, a description, associated environment variables,[6], and possible file extensions.

The last two entries in Table 1 are special cases, where the paths and environment variables depend on the name of the program: the variable name is constructed by converting the program name to upper case, and then appending INPUTS.

The environment variables are set by default in the configuration file texmf.cnf. It is only when you want to override one or more of the values specified in that file that you might want to set them explicitly in your execution environment.

Note that the "--format" and "--path" options are mutually exclusive.

--mode=*string*
Set the mode name to "*string*"; this only affects "gf" and "pk" lookups. No default: any mode will be found.

--must-exist
Do everything possible to find the files, notably including searching the disk. By default, only the ls-R database is checked, in the interest of efficiency.

--path=*string*
Search along the path "*string*" (colon-separated as usual), instead of guessing the search path from the filename. "//" and all the usual expansions are supported. The options "--path" and "--format" are mutually exclusive.

--progname=*name*
Set the program name to "*name*". This can affect the search paths via the "*.progname*" feature in configuration files. The default is "kpsewhich".

--show-path=*name*
shows the path used for file lookups of file type "*name*". Either a filename extension (".pk", ".vf", etc.) or a name can be used, just as with "--format" option.

--debug=*num*
sets the debugging options to "*num*".

6.2.3 Examples of use

Let us now have a look at Kpathsea in action.

```
>> kpsewhich  article.cls
/usr/texmf/tex/latex/base/article.cls
```

We are looking for the file article.cls. Since the ".cls" suffix is unambiguous we do not need to specify that we want to look for a file of type "tex" (TEX source file directories). We find it in the subdirectory tex/latex/base below the "TEXMF" root directory. Similarly, all of the following are found without problems thanks to their unambiguous suffix.

```
>> kpsewhich array.sty
   /usr/texmf/tex/latex/tools/array.sty
>> kpsewhich latin1.def
   /usr/texmf/tex/latex/base/latin1.def
>> kpsewhich size10.clo
   /usr/texmf/tex/latex/base/size10.clo
>> kpsewhich small2e.tex
   /usr/texmf/tex/latex/base/small2e.tex
>> kpsewhich tugboat.bib
   /usr/texmf/bibtex/bib/beebe/tugboat.bib
```

The last item is a BIBTEX bibliography database for *TUGBoat* articles.

```
>> kpsewhich cmr10.pk
```

Font bitmap glyph files of type .pk are used by display programs like dvips and xdvi. Nothing is returned in this case since there are no pre-generated Computer Modern ".pk" files on our system (since we use the Type1 versions on the CD-ROM).

```
>> kpsewhich ecrm1000.pk
/usr/texmf/fonts/pk/ljfour/jknappen/
...                   ec/ecrm1000.600pk
```

For the extended Computer Modern files we had to generate ".pk" files, and since the default METAFONT mode on our installation is ljfour with a

[6] You can find definitions for these environment variables in the file texmf.cnf (page 37)

base resolution of 600 dpi (dots per inch), this instantiation is returned.

```
>> kpsewhich -dpi=300 ecrm1000.pk
```

In this case, when specifying that we are interested in a resolution of 300dpi (`-dpi=300`) we see that no such font is available on the system. In fact, a program like dvips or xdvi would go off and actually build the `.pk` files at the required resolution using the script mktexpk.

Next we turn our attention to dvips's header and configuration files. We first look at one of the commonly used files, the general prolog `tex.pro` for TEX support, before turning our attention to the generic configuration file (`config.ps`) and the PostScript font map `psfonts.map`. As the "`.ps`" suffix is ambiguous we have to specify explicitly which type we are considering ("`dvips config`") for the file `config.ps`.

```
>> kpsewhich tex.pro
   /usr/texmf/dvips/base/tex.pro
>> kpsewhich --format="dvips config" config.ps
   /usr/texmf/config/config.ps
>> kpsewhich psfonts.map
   /usr/texmf/dvips/base/psfonts.map
```

We now take a closer look at the URW Times PostScript support files. The name for these in Berry's font naming scheme is "`utm`". The first file we look at is the configuration file, which contains the name of the map file:

```
>> kpsewhich --format="dvips config" config.utm
/usr/texmf/dvips/psnfss/config.utm
```

The contents of that file is

```
  p +utm.map
```

which points to the file `utm.map`, which we want to locate next.

```
>> kpsewhich --format="dvips config" utm.map
   /usr/texmf/dvips/psnfss/utm.map
```

This map file defines the file names of the Type1 PostScript fonts in the URW collection. Its contents look like (we only show some of the lines):

```
  utmb8r  NimbusRomNo9L-Medi     ... <utmb8a.pfb
  utmbi8r NimbusRomNo9L-MediItal... <utmbi8a.pfb
  utmr8r  NimbusRomNo9L-Regu     ... <utmr8a.pfb
  utmri8r NimbusRomNo9L-ReguItal... <utmri8a.pfb
  utmbo8r NimbusRomNo9L-Medi     ... <utmb8a.pfb
  utmro8r NimbusRomNo9L-Regu     ... <utmr8a.pfb
```

Let us, for instance, take the Times Regular instance `utmr8a.pfb` and find its position in the `texmf` directory tree by using a search for Type1 font files:

```
>> kpsewhich utmr8a.pfb
    /usr/texmf/fonts/type1/
...                 urw/utm/utmr8a.pfb
```

It should be evident from these few examples how you can easily locate the whereabouts of a given file. This is especially important if you suspect that the wrong version of a file is picked up somehow, since kpsewhich will show you the first file encountered.

6.2.4 Debugging actions

Sometimes it is necessary to investigate how a program resolves file references. To make this feasible in a convenient way, Kpathsea offers various debug levels:

- 1 `stat` calls (file tests). When running with an up-to-date `ls-R` database this should almost give no output.
- 2 References to hash tables (like `ls-R` database, map files, configuration files).
- 4 File open and close operations.
- 8 General path information for file types searched by Kpathsea. This is useful to find out where a particular path for the file was defined.
- 16 Directory list for each path element (only relevant for searches on disk).
- 32 File searches.

A value of `-1` will set all the above options; in practice you will probably always use these levels if you need any debugging.

Similarly, with the dvips program, by setting a combination of debug switches, one can follow in detail where files are being picked up from. Alternatively, when a file is not found, the debug trace shows in which directories the program looks for the given file, so that one can get an indication what the problem is.

Generally speaking, as most programs call the Kpathsea library internally, one can select a debug option by using the `KPATHSEA_DEBUG` environment variable, and setting it to (a combination of) values as described in the above list.

Let us consider, as an example, a small LATEX source file, `hello-world.tex`, which contains the following input.

```
\documentclass{article}
\begin{document}
Hello World!
\end{document}
```

This little file uses only the font `cmr10`, so let us look how dvips prepares the PostScript file (we want

to use the Type1 version of the Computer Modern fonts, hence the option `-Pcms`).

```
>> dvips -d4100 hello-world -Pcms -o
```

In this case we have combined dvips's debug class 4 (font paths) with Kpathsea's path element expansion (see dvips Reference Manual, `texmf/doc/html/dvips/dvips_toc.html`). The output (slightly rearranged) appears in Figure 5. dvips starts by locating its working files. First, `texmf.cnf` is found, which gives the definitions of the search paths for the other files, then the file database `ls-R` (to optimize file searching) and the file `aliases`, which makes it possible to declare several names (e.g., a short DOS-like "8.3" and a more natural longer version) for the same file. Then dvips goes on to find the generic configuration file `config.ps` before looking for the customization file `.dvipsrc` (which, in this case is *not found*). Finally, dvips locates the config file for the Computer Modern PostScript fonts `config.cms` (this was initiated with the `-Pcms` option on the dvips command). This file contains the list of the "map" files which define the relation between the TEX, PostScript and file system names of the fonts.

```
>> more /usr/texmf/dvips/cms/config.cms
   p +ams.map
   p +cms.map
   p +cmbkm.map
   p +amsbkm.map
```

dvips thus goes on to find all these files, plus the generic map file `psfonts.map`, which is always loaded (it contains declarations for commonly used PostScript fonts; see the last part of Section 6.2.3 for more details about PostScript map file handling).

At this point dvips identifies itself to the user...

```
This is dvips 5.78 Copyright 1998 Radical Eye...
```

then goes on to look for the prolog file `texc.pro`.

After having found the file in question, dvips outputs date and time, and informs us that it will generate the file `hello-world.ps`, then that it needs the font file `cmr10`, and that the latter is declared as "resident":

```
TeX output 1998.02.26:1204' -> hello-world.ps
Defining font () cmr10 at 10.0pt
Font cmr10 <CMR10> is resident.
```

Now the search (Figure 8) starts for `cmr10.tfm`, which is found, then a few more prolog files (not shown) are referenced, and finally the Type1 instance `cmr10.pfb` of the font is located and included in the output file (see last line).

6.3 Runtime options

Another of the nice features of Web2c 7.3 is its possibility to control a number of memory parameters (in particular, array sizes) via the runtime file `texmf.cnf` read by Kpathsea. The listing of `texmf.cnf` is shown in Section 7, starting on page 37; the settings of all parameters can be found in Part 3 of that file. The more important control variables are:

`main_memory` Total words of memory available, for TEX, METAFONT and MetaPost. You must make a new format file for each different setting. For instance, you could generate a "huge" version of TEX, and call the format file `hugetex.fmt`. Using the standard way of specifying the program name used by Kpathsea, the particular value of the `main_memory` variable will then be read from `texmf.cnf` (compare the generic value and the "huge" one instantiated by `hugetex`, etc.).

`extra_mem_bot` Extra space for "large" TEX data structures: boxes, glue, breakpoints, etc. Especially useful if you use PICTEX.

`font_mem_size` Number of words for font information available for TEX. This is more or less the total size of all TFM files read.

`hash_extra` Additional space for the hash table of control sequence names. Approximately 10,000 control sequences can be stored in the main hash table; if you have a large book with numerous cross-references, this might not be enough. You can see that both the hugetex and pdflatex program invocations ask for an extra 15,000 control sequences (the default value of `hash_extra` is zero).

Of course, this facility is no substitute for truly dynamic arrays and memory allocation, but since this is extremely difficult to implement in present TEX, these runtime parameters provide a practical compromise allowing some flexibility.

```
debug:start search(file=texmf.cnf, must_exist=1, find_all=1,
  path=.:/usr/local/bin/texlive:/usr/local/bin:
       /usr/local/bin/texmf/web2c:/usr/local:
       /usr/local/texmf/web2c:/.:/./teTeX/TeX/texmf/web2c:).
kdebug:start search(file=ls-R, must_exist=1, find_all=1,
  path=~/tex:/usr/local/texmf).
kdebug:search(ls-R) =>/usr/local/texmf/ls-R
kdebug:start search(file=aliases, must_exist=1, find_all=1,
  path=~/tex:/usr/local/texmf).
kdebug:search(aliases) => /usr/local/texmf/aliases
kdebug:start search(file=config.ps, must_exist=0, find_all=0,
  path=.:~/tex:!!/usr/local/texmf/dvips//).
kdebug:search(config.ps) => /usr/local/texmf/dvips/config/config.ps
kdebug:start search(file=/root/.dvipsrc, must_exist=0, find_all=0,
  path=.:~/tex:!!/usr/local/texmf/dvips//).
search(file=/home/goossens/.dvipsrc, must_exist=1, find_all=0,
  path=.:~/tex/dvips//:!!/usr/local/texmf/dvips//).
kdebug:search($HOME/.dvipsrc) =>
kdebug:start search(file=config.cms, must_exist=0, find_all=0,
  path=.:~/tex/dvips//:!!/usr/local/texmf/dvips//).
kdebug:search(config.cms)
=>/usr/local/texmf/dvips/cms/config.cms
```

Figure 5: Finding configuration files

```
kdebug:start search(file=texc.pro, must_exist=0, find_all=0,
  path=.:~/tex/dvips//:!!/usr/local/texmf/dvips//:
       ~/tex/fonts/type1//:!!/usr/local/texmf/fonts/type1//).
kdebug:search(texc.pro) => /usr/local/texmf/dvips/base/texc.pro
```

Figure 6: Finding the prolog file

```
kdebug:start search(file=cmr10.tfm, must_exist=1, find_all=0,
  path=.:~/tex/fonts/tfm//:!!/usr/local/texmf/fonts/tfm//:
       /var/tex/fonts/tfm//).
kdebug:search(cmr10.tfm) => /usr/local/texmf/fonts/tfm/public/cm/cmr10.tfm
kdebug:start search(file=texps.pro, must_exist=0, find_all=0,
   ...
<texps.pro>
kdebug:start search(file=cmr10.pfb, must_exist=0, find_all=0,
  path=.:~/tex/dvips//:!!/usr/local/texmf/dvips//:
       ~/tex/fonts/type1//:!!/usr/local/texmf/fonts/type1//).
kdebug:search(cmr10.pfb) => /usr/local/texmf/fonts/type1/public/cm/cmr10.pfb
<cmr10.pfb>[1]
```

Figure 7: Finding the font file

```
kdebug:start search(file=cmr10.tfm, must_exist=1, find_all=0,
  path=.:~/tex/fonts/tfm//:!!/usr/texmf/fonts/tfm//:
       /var/tex/fonts/tfm//).
kdebug:search(cmr10.tfm) => /usr/texmf/fonts/tfm/public/cm/cmr10.tfm
kdebug:start search(file=texps.pro, must_exist=0, find_all=0,
   ...
<texps.pro>
kdebug:start search(file=cmr10.pfb, must_exist=0, find_all=0,
  path=.:~/tex/dvips//:!!/usr/texmf/dvips//:
       ~/tex/fonts/type1//:!!/usr/texmf/fonts/type1//).
kdebug:search(cmr10.pfb) => /usr/texmf/fonts/type1/public/cm/cmr10.pfb
<cmr10.pfb>[1]
```

Figure 8: Finding the Type 1 font file

Table 1: Kpathsea file types

Name	*Description*	*Variables*	*Suffixes*
afm	Adobe font metrics	AFMFONTS	.afm
base	Metafont memory dump	MFBASES, TEXMFINI	.base
bib	BibTeX bibliography source	BIBINPUTS, TEXBIB	.bib
	bitmap fonts	GLYPHFONTS, TEXFONTS	
bst	BibTeX style files	BSTINPUTS	.bst
cnf	Runtime configuration files	TEXMFCNF	.cnf
dvips config	dvips configuration files, e.g., config.ps and psfonts.map	TEXCONFIG	.map
fmt	TeX memory dump	TEXFORMATS, TEXMFINI	.fmt, .efmt, .efm
gf	generic font bitmap	GFFONTS	.gf
graphic/figure	Encapsulated PostScript figures	TEXPICTS, TEXINPUTS	.eps, .epsi
ist	makeindex style files	TEXINDEXSTYLE, INDEXSTYLE	.ist
ls-R	Filename databases	TEXMFDBS	
map	Fontmaps	TEXFONTMAPS	.map
mem	MetaPost memory dump	MPMEMS, TEXMFINI	.mem
mf	Metafont source	MFINPUTS	.mf
mfpool	Metafont program strings	MFPOOL, TEXMFINI	.pool
mft	MFT style file	MFTINPUTS	.mft
	miscellaneous fonts	MISCFONTS	
mp	MetaPost source	MPINPUTS	.mp
mppool	MetaPost program strings	MPPOOL, TEXMFINI	.pool
MetaPost support	MetaPost support files, used by DMP	MPSUPPORT	
ocp	Ω compiled process files	OCPINPUTS	.ocp
ofm	Ω font metrics	OFMFONTS, TEXFONTS	.ofm, .tfm
opl	Ω property lists	OPLFONTS, TEXFONTS	.opl
otp	Ω translation process files	OTPINPUTS	.otp
ovf	Ω virtual fonts	OVFFONTS, TEXFONTS	.ovf
ovp	Ω virtual property lists	OVPFONTS, TEXFONTS	.ovp
pk	packed bitmap fonts	*program*FONTS (*program* being XDVI, etc.), PKFONTS, TEXPKS, GLYPHFONTS, TEXFONTS	.pk
PostScript header	downloadable PostScript	TEXPSHEADERS, PSHEADERS	.pro, .enc
tex	TeX source	TEXINPUTS	.tex, .cls, .sty, .clo, .def
TeX system documentation	Documentation files for the TeX system	TEXDOCS	
TeX system sources	Source files for the TeX system	TEXSOURCES	
texpool	TeX program strings	TEXPOOL, TEXMFINI	.pool
tfm	TeX font metrics	TFMFONTS, TEXFONTS	.tfm
Troff fonts	Troff fonts, used by DMP	TRFONTS	
truetype fonts	TrueType outline fonts	TTFONTS	.ttf, .ttc
type1 fonts	Type 1 PostScript outline fonts	T1FONTS, T1INPUTS, TEXPSHEADERS, DVIPSHEADERS	.pfa, .pfb
type42 fonts	Type 42 PostScript outline fonts	T42FONTS	
vf	virtual fonts	VFFONTS, TEXFONTS	.vf
web2c files	Web2c support files	WEB2C	
other text files	text files used by 'foo'	FOOINPUTS	
other binary files	binary files used by 'foo'	FOOINPUTS	

7 The texmf.cnf file

```
% TeX Live texmf.cnf
%  Part 1: Search paths and directories.
%
% You can set an environment variable to override TEXMF if you're testing
% a new TeX tree, without changing anything else.
%
% You may wish to use one of the $SELFAUTO... variables here so TeX will
% find where to look dynamically.  See the manual and the definition
% below of TEXMFCNF.

% The main tree, which must be mentioned in $TEXMF, below:
TEXMFMAIN = $SELFAUTOPARENT/texmf

% A place for local additions to a "standard" texmf tree.
TEXMFLOCAL = $SELFAUTOPARENT/texmf-local

% User texmf trees can be catered for like this...
HOMETEXMF=$HOME/texmf

% A place where texconfig stores modifications (instead of the TEXMFMAIN
% tree).  texconfig relies on the name, so don't change it.
VARTEXMF = $SELFAUTOPARENT/texmf-var

% Now, list all the texmf trees. If you have multiple trees,
% use shell brace notation, like this:
%   TEXMF = {$HOMETEXMF,!!$VARTEXMF,!!$TEXMFLOCAL,!!$TEXMFMAIN}
% The braces are necessary.
TEXMF = {$HOMETEXMF,$TEXMFLOCAL,!!$VARTEXMF,!!$TEXMFMAIN}

% The system trees.  These are the trees that are shared by all the users.
SYSTEXMF = $TEXMF

% Where generated fonts may be written.  This tree is used when the sources
% were found in a system tree and either that tree wasn't writable, or the
% varfonts feature was enabled in MT_FEATURES in mktex.cnf.
VARTEXFONTS = /var/tmp/texfonts

% Where to look for ls-R files.  There need not be an ls-R in the
% directories in this path, but if there is one, Kpathsea will use it.
TEXMFDBS = $TEXMF;$VARTEXFONTS

% It may be convenient to define TEXMF like this:
%   TEXMF = {$HOMETEXMF,!!$TEXMFLOCAL,!!$TEXMFMAIN,$HOME}
% which allows users to set up entire texmf trees, and tells TeX to
% look in places like ~/tex and ~/bibtex.  If you do this, define TEXMFDBS
% like this:
%   TEXMFDBS = $HOMETEXMF;$TEXMFLOCAL;$TEXMFMAIN;$VARTEXFONTS
% or mktexlsr will generate an ls-R file for $HOME when called, which is
% rarely desirable.  If you do this you'll want to define SYSTEXMF like
% this:
%   SYSTEXMF = $TEXMFLOCAL;$TEXMFMAIN
% so that fonts from a user's tree won't escape into the global trees.
%
% On some systems, there will be a system tree which contains all the font
% files that may be created as well as the formats.  For example
%   VARTEXMF = /var/lib/texmf
% is used on many Linux systems.  In this case, set VARTEXFONTS like this
```

```
%   VARTEXFONTS = $VARTEXMF/fonts
% and do not mention it in TEXMFDBS (but _do_ mention VARTEXMF).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Usually you will not need to edit any of the other variables in part 1. %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% WEB2C is for Web2C specific files.  The current directory may not be
% a good place to look for them.
WEB2C = $TEXMF/web2c

% TEXINPUTS is for TeX input files -- i.e., anything to be found by \input
% or \openin, including .sty, .eps, etc.

% LaTeX-specific macros are stored in latex.
TEXINPUTS.latex = .;$TEXMF/tex/{latex,generic,}//
TEXINPUTS.hugelatex = .;$TEXMF/tex/{latex,generic,}//

% Fontinst needs to read afm files.
TEXINPUTS.fontinst = .;$TEXMF/{tex{/fontinst,},fonts/afm}//

% Plain TeX.  Have the command tex check all directories as a last
% resort, we may have plain-compatible stuff anywhere.
TEXINPUTS.tex = .;$TEXMF/tex/{plain,generic,}//
% other plain-based formats
TEXINPUTS.amstex = .;$TEXMF/tex/{amstex,plain,generic,}//
TEXINPUTS.ftex = .;$TEXMF/tex/{formate,plain,generic,}//
TEXINPUTS.texinfo = .;$TEXMF/tex/{texinfo,plain,generic,}//
TEXINPUTS.eplain  = .;$TEXMF/tex/{eplain,plain,generic,}//
TEXINPUTS.jadetex = .;$TEXMF/tex/{jadetex,generic,plain,}//
TEXINPUTS.pdfjadetex = .;$TEXMF/{pdftex,tex}/{jadetex,generic,plain,}//

% e-TeX.
TEXINPUTS.elatex   = .;$TEXMF/{etex,tex}/{latex,generic,}//
TEXINPUTS.etex     = .;$TEXMF/{etex,tex}/{generic,plain,}//

% PDFTeX.  This form of the input paths is borrowed from teTeX.  A certain
% variant of TDS is assumed here, unaffected by the build variables.
TEXINPUTS.pdftexinfo = .;$TEXMF/{pdftex,tex}/{texinfo,plain,generic,}//
TEXINPUTS.pdflatex = .;$TEXMF/{pdftex,tex}/{latex,generic,}//
TEXINPUTS.pdftex   = .;$TEXMF/{pdftex,tex}/{plain,generic,}//
TEXINPUTS.pdfelatex = .;$TEXMF/{pdfetex,pdftex,etex,tex}/{latex,generic,}//
TEXINPUTS.pdfetex   = .;$TEXMF/{pdfetex,pdftex,etex,tex}/{plain,generic,}//

% Omega.
TEXINPUTS.lambda = .;$TEXMF/{omega,tex}/{lambda,latex,generic,}//
TEXINPUTS.omega = .;$TEXMF/{omega,tex}/{plain,generic,}//

% Context macros by Hans Hagen:
TEXINPUTS.context   = .;$TEXMF/{pdfetex,pdftex,etex,tex}/{context,plain,generic,}//

% cstex, from Petr Olsak
TEXINPUTS.cslatex = .;$TEXMF/tex/{cslatex,csplain,latex,generic,}//
TEXINPUTS.csplain = .;$TEXMF/tex/{csplain,plain,generic,}//
TEXINPUTS.pdfcslatex = .:$TEXMF/{pdftex,tex}/{cslatex,csplain,latex,generic,}//
TEXINPUTS.pdfcsplain = .:$TEXMF/{pdftex,tex}/{csplain,plain,generic,}//

% Polish
TEXINPUTS.platex  = .;$TEXMF/tex/{platex,latex,generic,}//
TEXINPUTS.pdfmex   = .;$TEXMF/{pdftex,tex}/{mex,plain,generic,}//
TEXINPUTS.mex   = .;$TEXMF/tex/{mex,plain,generic,}//
```

```
117
118 % french
119 TEXINPUTS.frtex   = .;$TEXMF/{mltex,tex}/{plain,generic,}//
120 TEXINPUTS.frlatex = .;$TEXMF/{mltex,tex}/{frlatex,latex,generic,}//
121
122 % MLTeX
123 TEXINPUTS.mltex = .;$TEXMF/{mltex,tex}/{plain,generic,}//
124 TEXINPUTS.mllatex = .;$TEXMF/{mltex,tex}/{latex,generic,}//
125
126 % odd formats  needing their own paths
127 TEXINPUTS.lollipop = .;$TEXMF/tex/{lollipop,generic,plain,}//
128 TEXINPUTS.lamstex  = .;$TEXMF/tex/{lamstex,generic,plain,}//
129
130 % Earlier entries override later ones, so put this last.
131 TEXINPUTS = .;$TEXMF/tex/{generic,}//
132
133 % Metafont, MetaPost inputs.
134 MFINPUTS = .;$TEXMF/metafont//;{$TEXMF/fonts,$VARTEXFONTS}/source//
135 MPINPUTS = .;$TEXMF/metapost//
136
137 % Dump files (fmt/base/mem) for vir{tex,mf,mp} to read (see web2c/INSTALL),
138 % and string pools (.pool) for ini{tex,mf,mp}.  It is silly that we have six
139 % paths and directories here (they all resolve to a single place by default),
140 % but historically ...
141 TEXFORMATS = .;$TEXMF/web2c
142 MFBASES = .;$TEXMF/web2c
143 MPMEMS = .;$TEXMF/web2c
144 TEXPOOL = .;$TEXMF/web2c
145 MFPOOL = .;$TEXMF/web2c
146 MPPOOL = .;$TEXMF/web2c
147
148 % Device-independent font metric files.
149 VFFONTS = .;$TEXMF/fonts/vf//
150 TFMFONTS = .;{$TEXMF/fonts,$VARTEXFONTS}/tfm//
151
152 % The $MAKETEX_MODE below means the drivers will not use a cx font when
153 % the mode is ricoh.  If no mode is explicitly specified, kpse_prog_init
154 % sets MAKETEX_MODE to /, so all subdirectories are searched.  See the manual.
155 % The modeless part guarantees that bitmaps for PostScript fonts are found.
156 PKFONTS = .;{$TEXMF/fonts,$VARTEXFONTS}/pk/{$MAKETEX_MODE,modeless}//
157
158 % Similarly for the GF format, which only remains in existence because
159 % Metafont outputs it (and MF isn't going to change).
160 GFFONTS = .;$TEXMF/fonts/gf/$MAKETEX_MODE//
161
162 % A backup for PKFONTS and GFFONTS. Not used for anything.
163 GLYPHFONTS = .;$TEXMF/fonts
164
165 % For texfonts.map and included map files used by mktexpk.
166 % See ftp://ftp.tug.org/tex/fontname.tar.gz.
167 TEXFONTMAPS = .;$TEXMF/fontname
168
169 % BibTeX bibliographies and style files.
170 BIBINPUTS = .;$TEXMF/bibtex/{bib,}//
171 BSTINPUTS = .;$TEXMF/bibtex/{bst,}//
172
173 % PostScript headers, prologues (.pro), encodings (.enc) and fonts;
174 % this is also where pdftex finds included figures files!
175
```

```
TEXPSHEADERS.pdflatex   = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdfelatex  = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdftexinfo = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdfcslatex = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdfcsplain = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdfetex    = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdfjadetex = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdfmex     = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdftex     = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.pdftexinfo = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.cont-de    = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.cont-en    = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.cont-nl    = .;$TEXMF/{tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS.context    = .;$TEXMF/{etex,tex,pdftex,dvips,fonts/type1}//
TEXPSHEADERS = .;$TEXMF/{dvips,fonts/type1,pdftex}//

% PostScript Type 1 outline fonts.
T1FONTS = .;$TEXMF/fonts/type1//

% PostScript AFM metric files.
AFMFONTS = .;$TEXMF/fonts/afm//

% TrueType outline fonts.
TTFONTS = .;$TEXMF/fonts/truetype//

% Type 42 outline fonts.
T42FONTS = .;$TEXMF/fonts/type42//

% A place to puth everything that doesn't fit the other font categories.
MISCFONTS = .;$TEXMF/fonts/misc//

% Dvips' config.* files (this name should not start with 'TEX'!).
TEXCONFIG = .;$TEXMF/dvips//

% Makeindex style (.ist) files.
INDEXSTYLE = .;$TEXMF/makeindex//

% Used by DMP (ditroff-to-mpx), called by makempx -troff.
TRFONTS = /usr/lib/font/devpost
MPSUPPORT = .;$TEXMF/metapost/support

% For xdvi to find mime.types and .mailcap, if they do not exist in
% $HOME.  These are single directories, not paths.
% (But the default mime.types, at least, may well suffice.)
MIMELIBDIR = c:/TeX/etc
MAILCAPLIBDIR = c:/TeX/etc

% TeX documentation and source files, for use with kpsewhich.
TEXDOCS = .;$TEXMF/doc//
TEXSOURCES = .;$TEXMF/source//

% Omega-related fonts and other files.  The odd construction for OFMFONTS
% makes it behave in the face of a definition of TFMFONTS.  Unfortunately
% no default substitution would take place for TFMFONTS, so an explicit
% path is retained.
OFMFONTS = .;{$TEXMF/fonts,$VARTEXFONTS}/{ofm,tfm}//;$TFMFONTS
OPLFONTS = .;{$TEXMF/fonts,$VARTEXFONTS}/opl//
OVFFONTS = .;{$TEXMF/fonts,$VARTEXFONTS}/ovf//
OVPFONTS = .;{$TEXMF/fonts,$VARTEXFONTS}/ovp//
```

```
235 OTPINPUTS = .;$TEXMF/omega/otp//
236 OCPINPUTS = .;$TEXMF/omega/ocp//
237
238 %% TeX4ht utility, sharing files with TeX4ht
239 T4HTINPUTS = .;$TEXMF/tex4ht//
240
241 %% The mktex* scripts rely on KPSE_DOT. Do not set it in the environment.
242 KPSE_DOT = .
243
244 % This definition isn't used from this .cnf file itself (that would be
245 % paradoxical), but the compile-time default in paths.h is built from it.
246 % The SELFAUTO* variables are set automatically from the location of
247 % argv[0], in kpse_set_progname.
248 %
249 % About the /. construction:
250 % 1) if the variable is undefined, we'd otherwise have an empty path
251 %    element in the compile-time path.  This is not meaningful.
252 % 2) if we used /$VARIABLE, we'd end up with // if VARIABLE is defined,
253 %    which would search the entire world.
254 %
255 % The TETEXDIR stuff isn't likely to be relevant unless you're using teTeX,
256 % but it doesn't hurt.
257 %
258 TEXMFCNF = .;{$SELFAUTOLOC,$SELFAUTODIR,$SELFAUTOPARENT}\
259 {,{/share,}/texmf{.local,}/web2c};c:/TeX/texmf/web2c
260
261
262
263 %  Part 2: Non-path options.
264
265 % Write .log/.dvi/etc. files here, if the current directory is unwritable.
266 % TEXMFOUTPUT = /tmp
267
268 % If a dynamic file creation fails, log the command to this file, in
269 % either the current directory or TEXMFOUTPUT.  Set to the
270 % empty string or  0  to avoid logging.
271 MISSFONT_LOG = missfont.log
272
273 % Set to a colon-separated list of words specifying warnings to suppress.
274 % To suppress everything, use TEX_HUSH = all; this is equivalent to
275 % TEX_HUSH = checksum:lostchar:readable:special
276 TEX_HUSH = none
277
278 % Enable system commands via \write18{...}?
279 shell_escape = f
280
281 % Allow TeX \openout/\openin on filenames starting with `.' (e.g., .rhosts)?
282 % a (any)        : any file can be opened.
283 % r (restricted) : disallow opening "dotfiles".
284 % p (paranoid)   : as 'r' and disallow going to parent directories, and
285 %                  restrict absolute paths to be under $TEXMFOUTPUT.
286 openout_any = p
287 openin_any = a
288 % Allow TeX, MF, and MP to parse the first line of an input file for
289 % the %&format construct.
290 parse_first_line = t
291
292 % Enable the mktex... scripts by default?  These must be set to 0 or 1.
293 % Particular programs can and do override these settings, for example
```

```
% dvips's -M option.  Your first chance to specify whether the scripts
% are invoked by default is at configure time.
%
% These values are ignored if the script names are changed; e.g., if you
% set DVIPSMAKEPK to 'foo', what counts is the value of the environment
% variable/config value 'FOO', not the 'MKTEXPK' value.
%
% MKTEXTEX = 0
% MKTEXPK = 0
% MKTEXMF = 0
% MKTEXTFM = 0
% MKOCP = 0
% MKOFM = 0

% What MetaPost runs to make MPX files.  This is passed an option -troff
% if MP is in troff mode.  Set to '0' to disable this feature.
MPXCOMMAND = makempx


%  Part 3: Array and other sizes for TeX (and Metafont and MetaPost).
%
% If you want to change some of these sizes only for a certain TeX
% variant, the usual dot notation works, e.g.,
% main_memory.hugetex = 20000000
%
% If a change here appears to be ignored, try redumping the format file.

% Memory. Must be less than 8,000,000 total.
%
% main_memory is relevant only to initex, extra_mem_* only to non-ini.
% Thus, have to redump the .fmt file after changing main_memory; to add
% to existing fmt files, increase extra_mem_*.  (To get an idea of how
% much, try \tracingstats=2 in your TeX source file;
% web2c/tests/memtest.tex might also be interesting.)
%
% To increase space for boxes (as might be needed by, e.g., PiCTeX),
% increase extra_mem_bot.
%
% For some xy-pic samples, you may need as much as 700000 words of memory.
% For the vast majority of documents, 60000 or less will do.
%
main_memory = 263000 % words of inimemory available; also applies to inimf&mp
extra_mem_top = 0    % extra high memory for chars, tokens, etc.
extra_mem_bot = 0    % extra low memory for boxes, glue, breakpoints, etc.

% Words of font info for TeX (total size of all TFM files, approximately).
font_mem_size = 200000

% Total number of fonts. Must be >= 50 and <= 2000 (without tex.ch changes).
font_max = 1000

% Extra space for the hash table of control sequences (which allows 10K
% names as distributed).
hash_extra = 0

% Max number of characters in all strings, including all error messages,
% help texts, font names, file names, control sequences.
% These values apply to TeX and MP.
pool_size = 125000
```

```
353
354 % Minimum pool space after TeX/MP's own strings; must be at least
355 % 25000 less than pool_size, but doesn't need to be nearly that large.
356 string_vacancies = 25000
357 max_strings = 15000               % max number of strings
358 pool_free = 5000                % min pool space left after loading .fmt
359
360 % Hyphenation trie. As distributed, the maximum is 65535; this should
361 % work unless 'unsigned short' is not supported or is smaller than 16
362 % bits.  This value should suffice for UK English, US English, French,
363 % and German (for example).  To increase, you must change
364 % 'ssup_trie_opcode' and 'ssup_trie_size' in tex.ch (and rebuild TeX);
365 % the trie will then consume four bytes per entry, instead of two.
366 %
367 % US English, German, and Portuguese: 30000.
368 % German: 14000.
369 % US English: 10000.
370 %
371 trie_size = 64000
372
373 % Buffer size.  TeX uses the buffer to contain input lines, but macro
374 % expansion works by writing material into the buffer and reparsing the
375 % line.  As a consequence, certain constructs require the buffer to be
376 % very large.  As distributed, the size is 50000; most documents can be
377 % handled within a tenth of this size.
378 buf_size = 50000
379
380 % These are Omega-specific.
381 ocp_buf_size = 20000        % character buffers for ocp filters.
382 ocp_stack_size = 10000        % stacks for ocp computations.
383 ocp_list_size = 1000        % control for multiple ocps.
384
385 % These work best if they are the same as the I/O buffer size, but it
386 % doesn't matter much.  Must be a multiple of 8.
387 dvi_buf_size = 16384    % TeX
388 gf_buf_size = 16384     % MF
389
390 % It's probably inadvisable to change these. At any rate, we must have:
391 % 45 < error_line      < 255;
392 % 30 < half_error_line < error_line - 15;
393 % 60 <= max_print_line;
394 % These apply to Metafont and MetaPost as well.
395 error_line = 79
396 half_error_line = 50
397 max_print_line = 79
398 stack_size = 300        % simultaneous input sources
399 save_size = 4000        % for saving values outside current group
400 param_size = 500        % simultaneous macro parameters
401 max_in_open = 15        % simultaneous input files and error insertions
402 hyph_size = 1000        % number of hyphenation exceptions, >610 and <32767
403 nest_size = 100               % simultaneous semantic levels (e.g., groups)
404
405 main_memory.context = 1100000
406 hash_extra.context = 25000
407 pool_size.context = 750000
408 string_vacancies.context = 45000
409 max_strings.context = 55000
410 pool_free.context = 47500
411 nest_size.context = 500
```

```
412 param_size.context = 1500
413 save_size.context = 5000
414 stack_size.context = 1500
415
416 main_memory.hugetex = 1100000
417 param_size.hugetex = 1500
418 stack_size.hugetex = 1500
419 hash_extra.hugetex = 15000
420 string_vacancies.hugetex = 45000
421 pool_free.hugetex = 47500
422 nest_size.hugetex = 500
423 save_size.hugetex = 5000
424 pool_size.hugetex = 500000
425 max_strings.hugetex = 55000
426
427 main_memory.hugelatex = 1100000
428 param_size.hugelatex = 1500
429 stack_size.hugelatex = 1500
430 hash_extra.hugelatex = 15000
431 string_vacancies.hugelatex = 45000
432 pool_free.hugelatex = 47500
433 nest_size.hugelatex = 500
434 save_size.hugelatex = 5000
435 pool_size.hugelatex = 500000
436 max_strings.hugelatex = 55000
437
438 main_memory.jadetex = 1500000
439 param_size.jadetex = 1500
440 stack_size.jadetex = 1500
441 hash_extra.jadetex = 50000
442 string_vacancies.jadetex = 45000
443 pool_free.jadetex = 47500
444 nest_size.jadetex = 500
445 save_size.jadetex = 5000
446 pool_size.jadetex = 500000
447 max_strings.jadetex = 55000
448
449 main_memory.pdfjadetex = 2500000
450 param_size.pdfjadetex = 1500
451 stack_size.pdfjadetex = 1500
452 hash_extra.pdfjadetex = 50000
453 string_vacancies.pdfjadetex = 45000
454 pool_free.pdfjadetex = 47500
455 nest_size.pdfjadetex = 500
456 save_size.pdfjadetex = 5000
457 pool_size.pdfjadetex = 500000
458 max_strings.pdfjadetex = 55000
459
460 main_memory.pdflatex = 1500000
461 param_size.pdflatex = 1500
462 stack_size.pdflatex = 1500
463 hash_extra.pdflatex = 15000
464 string_vacancies.pdflatex = 45000
465 pool_free.pdflatex = 47500
466 nest_size.pdflatex = 500
467 pool_size.pdflatex = 500000
468 save_size.pdflatex = 5000
469 max_strings.pdflatex = 55000
```