mk – a LaTeX maker

Wybo Dekker
wybo@servalys.nl

November 20, 2002

1 Introduction

MK is a Perl script that, in close collaboration with VPP [1] is helpful in the cyclic process of editing, viewing, and printing a LATEX document. Having an existing[2] LATEX document, say *main.tex*, you run MK on it by typing:

```
mk main
```

or, since *main* happens to be MK's default filename:

```
mk
```

Now, if *main.tex* is a valid LATEX source, MK compiles it, including any table of contents, indices, bibliography references, included files, and so on and VPP takes over and displays the resulting POSTSCRIPT or PDF output. When you leave the viewer you will see a prompt:

```
vpp command (h for help):
```

If you were satisfied with the displayed POSTSCRIPT or PDF output, you can now decide to print all or part of your document (see the section 'Page selection'), or you can simply quit by typing 'q'. On the other hand, if you decide that you want to change the source and have another try, you can edit the source by typing 'e' to get back to MK and (re)edit your source. After saving your work and leaving your editor, another compilation and display will be performed, based on the new source.

Essentially, MK uses the MAKE utility for the management of file dependencies, TEXI2PDF for compilation, GV (or any other viewer you like) for viewing, and CONTEXT's TEXEXEC for printing in various formats.

Currently, MK is only available for UNIX, but adaptation for MSDOS/WINDOWS should not be too complicated, as all software needed, including MAKE, is available on those operating systems.

[1] VPP (short for View and Print PostScript/PDF) is the subject of another article in this issue.

[2] We shall later explain what happens if you try to edit a non-existing file (see the section 'Locating the source').

2 How it works

When working on a LATEX document, the main activities — apart from thinking — consist of editing the sources, compiling with (PDF)LATEX, viewing the result (either a POSTSCRIPT or PDF file or LATEX errormessages) and, perhaps, printing the document or parts of it.

A LATEX document, as well as any other TEX document, may need several passes of compilation in order to fulfill all cross references and bibliography references, fix longtable calculations, and build the indices. When you compile manually, you'll have to keep track of the often abundant messages of LATEX to see if another compilation is needed. Most of that work can be taken out of your hands by using TEXI2PDF (or TEXI2DVI.) That is exactly what MK does.

However, although TEXI2PDF does look in the current directory for `\included` and `\inputed` files, it does not keep track of other files that you may have edited, such as style files, bibliography files, fontfiles, and so on, either in the current or in other directories. For that purpose, a recent contribution to CTAN by Tong Sun and Chris Beggy comes in handy: a makefile for MAKE that takes care of all of this, in cooperation with TEXI2PDF. However, this makefile needs to be told on what files your sources depend (apart from included files in the current directory.)

This problem is solved by the recent appearance of a new option for TEX: the **–recorder** option. This option tells TEX to maintain a *.fls* file that logs all file dependencies that TEX finds in the sources. This is how the start of the current document's *.fls* file, *main.fls*, looks like:

```
PWD /home/wybo/CVSWORK/mk
INPUT /tex/texmf/web2c/pdflatex.fmt
INPUT /tex/texmf/pdftex/config/pdftex.cfg
INPUT /home/wybo/CVSWORK/mk/main.tex
OUTPUT main.log
INPUT /texlive/texmf/tex/latex/base/article.cls
[ 110 similar lines follow... ]
```

Now here is how MK works (supposing *main.tex* to be the source):

1. if there is no file *main.fls* or if it is older than the source, LATEX is run to generate it.
2. *main.fls* is scanned for lines starting with INPUT and the files on those lines are saved for MAKE.
3. MAKE is executed.
4. if an error occurred, the log file is displayed, starting at the error location, skipping irrelevant lines, and stopping at most 20 lines later. The error message and its line number in the source are highlighted in color. The line number is remembered for the editor to start at. The user is finally asked whether he wants to quit, or to edit the source and recycle from 1. on.
5. if no error occurred, the PDF or POSTSCRIPT output is displayed, using VPP.
6. after the user has left the viewer (normally with 'q' or 'control-q') the user is asked (still running VPP) whether he wants to quit, or (re-)edit the source, or to print (parts of) the document.

3 Page selection

As said in the introduction, after a successful compilation and display of the resulting PDF or POSTSCRIPT output, the user is prompted with:

```
vpp command (h for help):
```

upon typing 'h' VPP displays examples of possible commands:

```
Examples of print commands:
  5       to print page 5
  5-      to print pages 5 through the end
  5-7     to print pages 5, 6 and 7
  -7      to print the first 7 pages
  5-7,19- to print pages 5, 6, 7 and 19 through the end
  a       to print the whole document
  a x3    to print 3 copies of the document
  x3      the same
  5 x3    to print 3 copies of page 5
  t       print the whole document twosided
  t 2-    print twosided starting at page 2
  b       to print the whole document as an a5 size booklet
  b -12   to print the first 12 pages as an a5 size booklet
Other commands:
  e       edit the tex source and rerun mk
  h       display this help
  ?       display this help
  q       quit
vpp command (h for help):
```

With these examples, no further explanation should be necessary, except that, when twosided ('t') or booklet ('b') printing is selected, printing will be performed in two shifts, one for the front side and one for the backside. Between the shifts, another prompt appears:

```
printer ready?  then turn stack and type return
```

You will have to arrange your printer such that, with the printed sides up, the first page printed will be at the bottom of the stack, and the last page printed will be on top. Normally you will then have your output come out the back of your printer. 'Turn the stack' then means: rotate it over the long side of the paper and feed it back into the printer for the other side to be printed.

For further information on VPP, look in its manpage by typing

```
vpp -h
```

or read the article on VPP elsewhere in this issue.

4 Locating the source

MK locates the LATEX source in several steps:

Table 1: MK options

	Options	Short	Defaults:
	––batch=*selection*	**–b**	
	––clean	**–c**	
	––Clean	**–C**	
⋆	**––print**	**–pr**	true
⋆	**––view**	**–vi**	true
⋆	**––ps**	**–ps**	false
⋆	**––quiet**	**–q**	true
	––rc=*rcfile*	**–r**	
	––edit=*file*	**–e**	
	––help	**–h**	
	––version	**–ve**	

1. If you supply no arguments, the file *main.tex* in the current directory is assumed.
2. If you supply an argument (say *myfile*,) MK adds a *.tex* extension if it isn't there and looks for *myfile.tex* in the current directory.
3. if *myfile.tex* is not found in the current directory, MK looks in the 'alternate directory' (say */Documents*) if you have defined one (see the section 'RC-files').
4. if the source was not found in /Documents, MK thinks that you may have a subdirectory *myfile* in /Documents where the source may live under the name *main.tex*
5. if that file is not there, MK now concludes that the source does not yet exist and reports this, telling at the saem time which files have been tried.
6. if you have defined a template file (see the section 'RC-files'), MK now gives you the opportunity to create a new LATEX source from that template. If you confirm MK's question, MK copies the template to the filename you supplied (or *main.tex* if you did not) and starts your editor with the newly created file.
7. finally, if all the above did not lead to a source file, MK dies.

5 Options

MK comes with several options. Table 1 shows an overview. Options are shown in logically identical pairs, with the full version in the first column and the minimum shorthand (without the parameters) in the second. Options marked with a star are boolean options. Default values are shown in the last column. You can set boolean options to false by prefixing the option with 'no', for example: **––noquiet** or **–noq**.

Before evaluating any options, MK will try to read a system rc-file, a user rc-file, and, finally an rc-file in the current directory. The default values for ⋆-marked options and for string options can be set in these files. See the section 'RC-files' for more information.

You can also set option defaults in an alias. For example:

```
alias mk='mk --noquiet'
```

––help Prints help information and lets you type 'm' to display the complete man page or anything else to quit.

––version Prints name and (cvs-)version and then quits.

––quiet Suppresses messages about the progress MK is making. This is the default.

––rc *rc-file* Read specified rc-file before processing. The contents of the rc-file may override options specified before the **––rc** option, therefore it is a good idea to have the habit of specifying the **––rc** option first.

––batch *printing command string* Prevents the **––print** option to interrogate the user about pages to be printed. Instead the document is printed according to the mandatory *printing command string*. Also sets viewing off. Thus the command

```
mk --batch '2-3 x3' test
```

prints 3 copies of pages 2 and 3 of *test.tex*, without viewing.

––clean Clean up (remove) all unnecessary files generated by LATEX and BIBTEX except for the PDF or POSTSCRIPT files.

––Clean Clean up (remove) all unnecessary files generated by LATEX and BIBTEX including the PDF or POSTSCRIPT files.

––print Present the print prompt. This is the default. This option is normally used to suppress the print prompt, for example when using MK from other scripts that generate LATEX documents that have only to be displayed or stored without even being displayed.

––ps Generate POSTSCRIPT version of document. The default is to generate a PDF document.

––view Run the file viewer. This is the default. This option is normally used to suppress starting the viewer, for example when using MK from other scripts that generate LATEX documents that have only to be printed.

––edit *file* Normally, MK lets you edit the main source file, but here you can specify another file to be edited instead. This is useful, for example, if you are are fixing a style file or another input file.

6 RC-files and customization

Unless the environment variable NORC has been set, three rc-files are executed, if they exist, before reading the command line options, in the following order:

1. `/etc/mkrc`: the system rc-file
2. `$HOME/.mkrc`: the user rc-file
3. `./.mkrc`: the local rc-file

You can use these rc-files to set the default values for the options, by setting the Perl variable named after the long version of the options. For example:

```
$quiet=1; # run in quiet mode
```

So if you usually like MK to work quietly, you can indicate so in your rc-file and change your mind in some cases by using the **––noquiet** (or perhaps **–noq**) option.

Other variables that can be set in the rc-files, and their default values, are:

`$skip_pattern = '';` can be set to a file wildcard pattern. Files matching this pattern on which the LATEX source file may depend will not be checked for changes. For example, if you use a write-protected TEX-tree in the directory *texlive* it makes sense to set `$skip_pattern='/^\/texlive/';`

`$altdir = '';` If `$altdir` is non-empty and a file to be compiled does not exist in the current directory, it will be given another try after prefixing it with the contents of `$altdir`. So if you like to have your LATEX file in */Documents/myfile.tex* you can set `$altdir` to `/Documents` and run MK from any directory with:

```
mk myfile
```

However, a directory like */Documents* does not make much sense if many of your LATEX documents do not consist of a single file, but are constituted of an ensemble of a main LATEX source and one or more `\included` and `\inputed` files such as graphics. You will then probably prefer to have s subdirectory in */Documents* for every LATEX document.

Therefore, if MK does not find *myfile.tex* in the alternate directory, it will assume that *myfile* is a subdirectory with a main LATEX source in it, called *main.tex*.

`$default = 'main';` This is the default for the basename of your LATEX document.

`$template = '';` Tells MK to give the opportunity to create a copy of this file when a non-existent source is requested.