Site Reports

TEX ACTIVITY AT SPRING DECUS

Barbara Beeton

The theme of the Languages and Tools SIG at the Spring DECUS meeting (New Orleans, May 27–31) was "Tools for Software Documentation". TEX and friends were very well represented, and the various presentations were met with considerable enthusiasm. The progam included the following:

- Patti Anklam, DEC, *Tools for software document production*, describing the extensive front-end input and file management system and TEX output interface through which the VAX/VMS 4.0 software documentation was generated, all 10,000 pages of it!
- Barry Ferris, Talaris, *Literate programming — Programs as works of literature*, describing the WEB system.
- David Spencer, Oregon Software, *Automating document production*, describing the techniques used to generate several different manuals from a single set of TEX-encoded files.
- Sam Whidden, AMS, *TEX: Typesetting for almost everyone, an overview*, describing in general terms the WEB/TEX/METAFONT system; and a follow-on Q&A session, assisted by Barbara Beeton.

The presentations by Patti Anklam and David Spencer illustrated the extremes of interfacing documentation writers to a production system.

The DEC approach insulates them entirely from the typesetting language: a file/code management system provides templates to cue the writers for the (generically identified) data and text elements required by the document design, and typesetting instructions (e.g. TEX codes) are inserted only later by a preprocessor; in fact, one early implementation of the system applied DEC Runoff codes, before the TEX macros were working, so that draft output could be obtained for proofreading with no delay. Flexibility in a typographic sense is somewhat limited, but the document design stresses (and enforces) logical consistency, while the TEXperts in the documentation group are very cooperative about adding new structures when they can be justified.

At Oregon Software, the writers actually key in the TEX codes; a question as to whether this was a temptation got the answer, "Well, yes — you sometimes see a writer running down the hall, waving a piece of paper, shouting 'Look at this neat thing i just found!' ".

But both groups agree: TEX is a powerful tool that permits them to generate quality documentation, in both content and appearance, with a great deal of flexibility, in a reasonable, and often limited, amount of time.

Looking toward the future, the editor of the DECUS Proceedings has inquired about making available a TEX header to produce copy in the appropriate format. Such a header will be devised here at AMS, put onto the DECUS distribution tapes (for use by authors who already have Plain TEX available), and introduced formally at a session at the Fall DECUS in Anaheim.

CMS TEX SITE REPORT

Alan Spragens
Stanford Linear Accelerator Center

This site report is rather overdue, considering that there has been an "official" VM/CMS version of TEX available for a year and a half. The current release level is 1.1; work on the 1.3 TEX is in progress—at least I'm collecting useful files and polling CMS sites for ideas and information. In this report I'll describe some of the history and peculiarities of the CMS port and some of the things we'd like to do with it in the future. We can always use help—this is definitely a group effort. I apologize to any contributors to the current state of CMS TEX whose names I have omitted, and for any inaccuracies I may have introduced.

The VM/CMS port of TEX is a combination of efforts of many people. WEB change files for preliminary versions of TEX were brought to the TUG meeting in August 1983 and given to David Fuchs who combined them into a definitive version on the VM system at Stanford Linear Accelerator Center. The original change files were supplied by Roger Chaffee of SLAC (now at Metaphor), Craig Platt of the University of Manitoba, and Peter Sih of IBM Palo Alto Scientific Center. Bernd Schulze of the University of Bonn also contributed his experience from bringing up a preliminary TEX under CMS. All of this work was based on preliminary work adapting TEX to MVS by Eagle Berns and Susan Plass of Stanford University.

Peter Sih generated the system for the distribution tape and contributed device support for several IBM printers. I expect this same software will work for the IBM 3820 printer which doesn't yet have VM support. Many valuable revisions and additions were made by Robert Creasy, including notes on how to use the IBM printer-support utilities. Alan Spragens at SLAC coordinates communication among the CMS sites.

In addition to all the TEX stuff, the current tape has **METAFONT** version −40.0 and a few associated programs. Thomas Denier, of the University of Pennsylvania, made a good start on a CMS change file for **METAFONT** and allowed me to include it with the caveat that it is work in progress. Pete Sih reworked that change file so that it would compile, and that's what went out with TEX 1.1. Since then, several people have made working CMS implementations. Tom Denier continued his work for a while, and sent some driver software for VT-100 terminals running through the Yale ASCII terminal interface. Klaus Thull, at IBM in Heidelberg, brought up version −40.0 and has done work with gray-scale fonts using various output devices including the IBM Sherpa (which is odd because it "writes white," causing various oddities to appear in letter shapes.) His **METAFONT** also makes output for IBM graphics terminals through the GDDM driver. The same version is running at Weizmann Institute, thanks to Sig Handelman. Bernd Schulze brought up **METAFONT** version 0 at the University of Bonn where it is now creating letter-forms and logos. Bernd has written a program with which he converted the Hershey fonts into **METAFONT** input. His TEX is in production, running output to an AGFA P400 LED printer and 3277GA softcopy display, and has produced some large mathematical papers and a full-size book. All these folks have kindly offered to share their work, so I hope we'll have some good **METAFONT** material on the next CMS TEX distribution tape.

The CMS TEX tape includes "load-and-go" modules for TEX and its friends, as well as all the source files and change files. The change files for other systems, as distributed on the generic tape are also there. It has `fmt` files for LATEX, AMS-TEX, and HPTEX and has a preliminary version of Art Ogawa's Psizzl macros, which were written at SLAC to format physics papers and theses. Peter Sih's device support for the IBM printers, including fonts, is distributed on separate tapes by DP Services, to keep the cost of all the tapes the same.

The CMS TEX has a few peculiarities worth noting here. First is the character translation business. The ASCII-to-EBCDIC conversion done at Stanford may not suit everyone. Of course, the ideal would be to have everything match *The TEXbook*, but EBCDIC terminals usually are missing some of the characters. So people with EBCDIC terminals will probably have to use characters that don't match in any case. Terminals lacking the backslash and brace characters like the 3277 are not fun; it helps to have the APL feature. Most IBM sites also use a lot of ASCII terminals and have some EBCDIC-to-ASCII translation facility. Those local translations will probably disagree with the Stanford ones. Translated back to ASCII, then, TEX still won't look right. Other than changing your ASCII terminal interface, there are at least four approaches to take: use the characters as they come on the tape; do a global edit on every source file on the tape; change some of the `catcodes` in a local file to input along with `PLAIN`; provide execs for setting CMS input and output translate tables with perhaps a profile for each terminal type. Pete Sih wrote an exec called `TEXTERM`, which is on the tape as an example of the fourth approach; at SLAC, we translate the offensive characters on all the source files. I doubt there is much disagreement among the translations aside from some of the non-alphanumerics, but a few of them are important TEX characters. I have received a number of well-considered opinions on this subject from CMS TEX sites, including comments that appeared in the TEXhax conference, which I hope will produce a better translation convention for the next CMS tape.

CMS TEX is invoked in pretty much the same way as any TEX, reading its arguments from the command line, as described in *The TEXbook*. Of course, this differs from normal CMS style in which spaces separate filenames from filetypes and so on. Also, Pascal/VS (at least the 2.1 version) has the unfortunate practice of picking up the tokenized, uppercase version of the CMS parameter list from the command line. That is, the command line is butchered before TEX sees it. For example, if you want to tell TEX to run a file in `\batchmode` without changing the file, so you invoke it with `tex \batchmode \input foo` you are liable to see TEX say `? Undefined control sequence \BATCHMO`. It's easy to avoid the problem with a front-end exec that puts the untokenized string in the console stack, and we've included one by Pete Sih (in EXEC2) on the tape. The only unfortunate consequence of this method is that TEX writes its "**" prompt before it reads the stacked string, and there is a pause during which a nervous user may type something, interrupting TeX.

Input files can be on any accessed disk and can be selected with a specification like `my.file.c` or with the default filetype of `tex` and filemode of * (first match in your search order) if only a filename is given. Files `quit.tex` (containing only an `\end`) and `null.tex` (containing a comment) are supplied to get TeX out of its "`can't find`..." loop. All output files are made on the A-disk. Log files have filetype `texlog`, which we use to avoid conflict with batch log files and such, although *The TeXbook* says `log`.

The `e` response to TeX's `?` prompt invokes XEDIT with the current line number in the console stack. This usually puts the current line on the error line, although the user's XEDIT profile could destroy the stack. Pete Sih contributed a small assembler routine that passes a program attention interrupt to TeX, invoking its `errorstop` routine, rather than interrupting CMS. Thus, the CMS TeX is pretty standard in spite of its living in an environment somewhat different from a typical ASCII system.

Although several people have done "core-image" dumps of TeX with the `plain.fmt` information incorporated, saving a ready-made, format-included module, some experimentation at SLAC showed that the CMS I/O can read the `fmt` file at runtime efficiently enough that there is little or no time saved by reading the larger, format-included module.

The `\write15` stream has been co-opted to send its argument to CMS via the Pascal/VS CMS procedure. This feature enables TeX users to run execs, XEDIT or other shared-segment programs, or send messages via CP/CMS messaging facilities during TeX processing.

The tape includes four utilities to interface TeX to the IBM 4250 electro-erosion printer and 3800 model 3 page printer, all of which were written by Pete Sih. The first utility, DVI4250, converts a DVI file into a format edible by the 4250. You still need the CDPF device driver that comes with the 4250. The second utility, DVI38PP does the same for the 3800 model 3. If you have a Sherpa (or APA6670), its current-release software has an interface to accept the 3800 data stream. The third utility, PXLCVT, converts the old **METAFONT**'s PXL fonts into 4250 format and 3800 format. The fourth utility, FONTPL, makes TeX-type property-list (PL) files from IBM font files of filetype FONT4250 and FONT38PP.

These utilities are also available on two auxiliary VM/CMS TeX tapes. Each tape comprises complete printer support for TeX on an IBM printer: one for the IBM 4250 electro-erosion printer, the other for the IBM APA6670 laser printer and 3800 model 3 page printer. On these tapes are the four utility programs previously described and the Computer Modern family of fonts, converted to the format accepted by the printer. The auxiliary tape for the 4250 is definitely recommended since 600 dot-per-inch PXL files are not part of the standard TeX Project font library. Of course, these considerations will be obviated by the new **METAFONT**.

As shown on the output device chart, a number of VM sites print TeX output on non-IBM printers. I would like to put more DVI-translators on the distribution tape. I'm looking forward to seeing Arthur Samuel's Imagen driver, which will use **METAFONT**'s GF files directly.

The next thing on the agenda, of course, is TeX 1.3 and the new **METAFONT**. Exactly when a new tape will come out is difficult to say, but I hope we'll be able to get something rolling in the next few months—maybe before or shortly after TUG meeting time. The memory scheme in TeX 1.3 should make for more efficient processing in VM's paged environment. Several people have inquired about making a shared-segment TeX, which would be nice. I believe that the current Pascal/VS is unable to produce re-entrant code; the current TeX module stores into itself. Nonetheless, I still inquire into the possibility at every opportunity; perhaps there is some way I don't know about to get Pascal/VS to allocate free storage at run time. I did get hold of an external procedure to link into a module that will enable Pascal/VS to use the untokenized CMS parameter list, so I hope we'll be able to avoid the `\BATCHMO` problem. Chris Thompson at Cambridge University has done some experiments manipulating the Pascal/VS stack in various ways which may produce a 20% or better improvement in TeX's execution time. Other improvements may be possible, and I hope we can implement a number of them this time around.

Report on Experience with TEX80

Hans Riesel
Royal Institute of Technology, Stockholm

Reason for this Report. In TUGboat 5 (1984) p. 11, Donald E. Knuth asks about some experience with TEX reached at the Royal Institute of Technology in Stockholm. This report has been written in order to give the author's personal answer to his question.

Project Description. TEX80 was introduced on a rather small scale at the Royal Institute of Technology in Stockholm in February 1982. The author of this report had at that time the privilige to attend a one week introductory course. My goal was to do the typesetting of a math book, *Prime Numbers and Computer Methods for Factorization*, Birkhäuser, Boston, 1985, pp. xvi+464. The work was finished in January 1985. The author has so far had practically no experience with later versions of TEX, so it may occasionally happen that the views expressed in this report are outdated by the development of more modern versions of TEX.

The computers used were various DEC-20 computers with a Versatec V80 as output unit. The Versatec has a resolution of 200 dot/inch and the driver magnifies everything by a factor of 1.301, so cmr10 is shown as cmr13. This output size is then photographically reduced before printing.

Useful Features of TEX. First I would like to mention some positive things with TEX. It is really great fun working with TEX—the author has no graphical or typographical education at all and has thus (mostly) experienced each page coming out of the Versatec printer as a very pleasant surprise.

I think that TEX's most valuable feature is the option for authors to write own makros. I have been taking advantage of this all the time, mainly in one of the following situations:

1. For details of graphical design, such as the indentation of the paragraph the reader is just looking at.
2. To circumvent bugs in TEX.
3. To postpone the solution of problems that could not be solved immediately, e.g. if "e.g." should be typeset in italics or in roman letters. In such a situation I just wrote a makro **\eg**, yielding e.g. and then I decided on the precise form later on. Having given the makro **\eg** its final form all e.g.'s appeared as decided in the next draft of the text. This turned out to be a very convenient way of working.
4. To aid people untrained in computer programming to use TEX. At the Institute for Numerical Analysis and Computing Science one of the secretaries took active interest in learning TEX and did a very fine job on typesetting reports. Some of the more difficult types of math formulas, however, did not come out too well, despite her efforts. In such situations the solution was to write a makro with parameters, which could easily be used by her to produce the wanted result.
5. To create complicated expressions such as

$$b_0+\cfrac{a_1}{b_1+\cfrac{a_2}{b_2+\cfrac{a_3}{b_3+\cdots+\cfrac{a_n}{b_n}}}}$$

The beautiful thing with a makro doing this is that TEX automatically expands the boxes involved, if needed. Thus by letting the parameters of the makro be themselves complicated expressions, things like

$$(1+x)^\mu=1+\cfrac{\mu x}{1+\cfrac{\frac{1(1-\mu)}{1\cdot2}x}{1+\cfrac{\frac{1(1+\mu)}{2\cdot3}x}{1+\cfrac{\frac{2(2-\mu)}{3\cdot4}x}{1+\cdots}}}}$$

can easily be typeset.

Also very important is the use of a format file to define the typographical format of your work (where to put page numbers, running heads etc.). If all such information is collected in a separate file, it is handy to introduce changes, if needed. An example of this is the following. I made the mistake to discuss my book with a graphical designer at a much too late stage of the work, but was partly saved by having put all formatting information in one place. Thus I could fairly easily redesign the output format—as a matter of fact, quite substantial re-formatting took me only a couple of hours to achieve.

One very important consequence of using TEX is that you as an author are working much more intimately with your text than you do by the conventional method, i.e. by typing your manuscript or letting your secretary do the typing and then sending it to a publisher. I think that the quality of the text is enormously improved by this way of working, including its intellectual content. In particular the troublesome proof reading, which earlier was done a long time after the work had been submitted for publication, and which was often permitted to take only a short time (i.e. had to be done during night hours) is now done in parallel with the other processing of the text.—I furthermore think that it is much easier to "sell" a work to a publisher when it is nicely presented instead of looking dull and ugly as typed pages with inserted hand-written formulas frequently do.

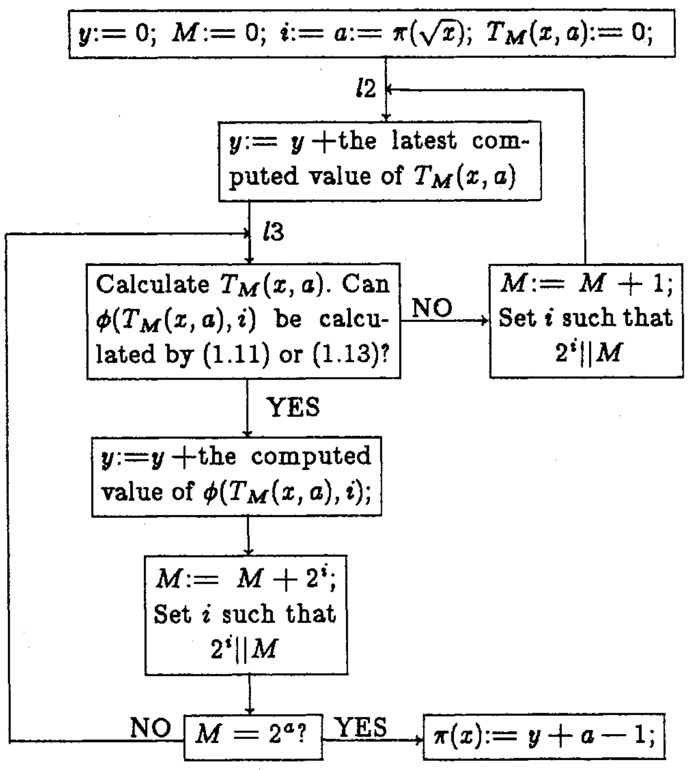
Some Limitations of TEX80. I made some attempts to draw flow-charts with TEX, but I have to advise against this. The best result I managed to achieve after about one week of work was the following:

I was quite discouraged by finding that I had insufficient patience to get rid of the small breaks which show up here and there in the flow-lines! (In the design of tables, however, where I encountered a similar problem, which simply *had* to be solved since my book contains some 100 pages of tables, I managed to find a remedy in the following macro

```
\def\norule{\noalign{\hrule height 0pt}}
```

which I wrote for the purpose.)

Four Professions in One. I here want to take the opportunity of pointing out one consequence of letting the author do the typesetting, a consequence of which I was not fully aware when I began using TEX three years ago. It is the fact that the author has to master, at least to some degree, three more professions, namely the profession of typographer, of graphic designer and of text editor. It does not


$y{:=}\ 0;\ M{:=}\ 0;\ i{:=}\ a{:=}\ \pi(\sqrt{x});\ T_M(x,a){:=}\ 0;$
$l2$
$y{:=}\ y+$the latest computed value of $T_M(x,a)$
$l3$
Calculate $T_M(x,a)$. Can $\phi(T_M(x,a),i)$ be calculated by (1.11) or (1.13)?
NO
$M{:=}\ M+1;$ Set i such that $2^i||M$
YES
$y{:=}y+$the computed value of $\phi(T_M(x,a),i)$;
$M{:=}\ M+2^i;$ Set i such that $2^i||M$
NO
$M=2^a?$
YES
$\pi(x){:=}\ y+a-1;$

suffice to know something about typography.—The graphic designer helps you to create beautiful pages by balancing the types chosen for the running heads, the chapter titles and the subheadings and so forth, with the rest of the text. And, finally, the text editor checks that there are no inconsistencies in the use of grammar, of notation and so on throughout the work, and also that the usage chosen conforms with "normal" usage in the field in question.

Criticism of TEX80. The Fonts. Since I have no education at all in typography or graphic design, I felt quite happy with the result I got from TEX during the first two years I was writing my book. I used the computer as an electronic scratchpad, produced my typeset pages and found everything nice. When the job was almost finished I hit upon the idea to discuss a draft of the text with a graphic designer. He looked at the pages and asked me how I had managed to get them this way. I answered: "Well, I looked at some books which I thought looked nice and tried to imitate them!" A long silence followed. Then the book designer said: "Maybe those books you have been looking at don't represent the best in graphic design!"—But he was very kind and helpful and gave me some valuable advice which I followed and which fixed things up quite a bit.

This graphical designer had never had the opportunity to look closer into computer created fonts before I showed his my pages. His general reflexion was: "With so many bad fonts around, the last thing needed is another bad font!" His main objection to cmr was that the characters look a little bit too crowded, the serifs being too close to some other part of the letter. This creates an impression of a rather heavy type. The heavy looks could, fortunately, be mitigated by simply increasing the baselineskip by one pt, which I did. (So I finally used cmr10 on 13 pt.)

The other major objection from the graphical designer concerned the use of slanted instead of italics. He found this habit rather bad.—Curiously enough, cmi9 looks much better than cmi10, so I actually used italics in the lists of references, which were typeset in ninepoint, but I had to maintain cmsl0 in the main text.

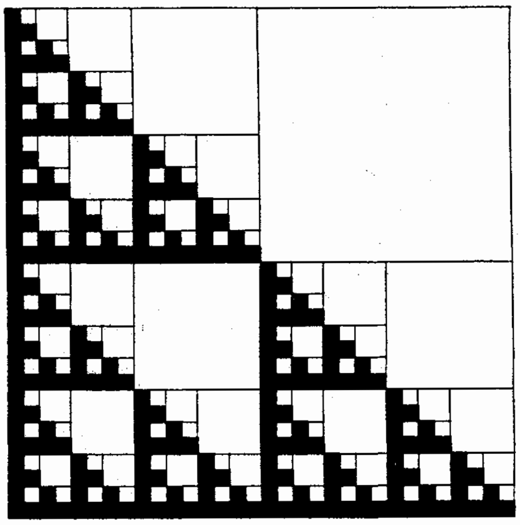
Furthermore, the spectrum of different fonts available turned out to be too limited for my purpose. I would have liked to have true small caps rather than having to use caps in a smaller font.—Touching the subject of fonts I have to inform the reader that there are three national characters in the Swedish alphabet, the å, ä and ö, which have to be incorporated before a font is ready for use in Swedish. Unfortunately the Swedish national characters replace some useful special characters on the terminal's keyboard, i.a. [and], and so *these* in turn have to be replaced by makros. The end result of this whole business is that we here at the Royal Institute have a slightly more complicated version of TEX than at Stanford and, worst of all, our TEX-files are incompatible with TEX-files in other countries. We thus have to send paper copies or films of camera ready texts rather than magnetic tapes when a text is to be reproduced abroad.

The Complexity of TEX. I think TEX, as being created by a computer scientist, is about as nice as it could be *for other computer scientists.* But for people ignorant of computer programming, TEX is much too difficult to learn to master. And, unfortunately, there does not seem to be a useful subset of reasonable complexity either—since TEX is built on the idea of boxes and glue, only the most trivial applications, such as writing letters, can be taught without explaining boxes and glue. But, perhaps this is how it has to be, since typesetting is evidently a much more complicated business than just using a typewriter. I have often found the reaction that TEX is too complicated, when trying to persuade colleagues at the Royal Institute to try it. The computer scientists learn it fast, but for other people the initial threshold often seems unsurmontable.

Future Prospects. Since the benefit is so large when an author is doing the typesetting himself or, at least, is supervising his secretary typesetting it, I think every effort should be made in order to achieve this mode of working. To try this idea I have made a couple of attempts to use TEX for papers in journals, one unsuccessful with a local journal (in Swedish) for math teachers, and one with *BIT,* which I found very rewarding. In this case I managed to persuade the editor to accept a camera ready copy of a paper of 18 pages. As a result, I was permitted to deliver the article about six months closer to its publication date than elsewise, a fact of which I took advantage by doing some late updating of the material in the article and in the list of references.—Unfortunately, however, the font used in *BIT* was not available in TEX80, so the typography of my paper contrasts somewhat with the rest of the material in the journal. (The same remark applies to this report, which had to be done in cmr instead of amr.) It also took me a much bigger effort to write the master file for the graphical format of *BIT* than one might expect an average author to be prepared to put in. But if master files with specs and with the appropriate fonts could be made easily available to authors of papers intended for the leading journals, I am quite sure that this way of working would be used more and thus would help to cut down production costs of math journals.

To link TEX with computer graphics is another important possibility. For my book I needed a picture of a certain fractal structure, which is shown here. This was done for me with the aid of TEX82 with a very short program written by a colleague of mine, Lennart Börjeson.

Finally, the reduction by TEX and similar systems of the very high costs involved in the typesetting of mathematical formulas, having so far prevented the publication of some highly specialized works with only a small market, will undoubtedly expand the domain of works, deemed economical to publish. This will clearly have a positive influence on the market of math books.—In this line of thought lies also the fact that works rejected by journals can now be published by their authors in a form that is at least typographically attractive.


Fractal structure, programmed by Lennart Börjeson using TEX82.

A TEX82 IMPLEMENTATION ON THE HP9000 SERIES 500.

Gregory Marriott
Thermodynamics Research Center
Texas A&M University

Here at TRC, one of our functions is to publish tables of thermodynamic properties of compounds. Until recently, the introductions to these publications were done using an IBM composer. Mathematical symbols were made using rub-on transfers. This made modification of the documents a long and drawn out process. When we first heard of TEX, it sounded like what we were looking for, but no version was available which could run on our computer.

Our computer is an HP9000 series 500. This is a 32-bit, 32-user system with HP-UX version 4.02. HP-UX is HP's version of UNIX, based on System III. We have 4 Mb of memory and 140 Mb of mass storage. Since we knew of nobody with an HP9000 implementation, I decided that I should try an implementation.

At the start, I had almost no experience with UNIX, none with Pascal, and I thought WEB was something spiders caught bugs in. (As it turned out, WEB caught several of **my** bugs!) I did, however, have the good fortune of having at my disposal a copy of *The WEB System of Structured Documentation*. This contained a description of WEB, but most important it included a step by step procedure for implementing TEX82. In a relatively short period of time, I learned enough WEB, Pascal and UNIX to bring up **tangle**. Shortly thereafter **weave** came to life. It all seemed easy. I hadn't run into a major problem. TEX looked like it was just a couple of days away.

Somewhere I found a copy of the TEX listing and started in on the index, in the section marked **system dependencies**. I spent the next couple of days making notes and a change file for **tangle**. When I ran **tangle** on **tex.web**, my hopes began to fade away. I discovered that I hadn't learned WEB as well as I thought. Many long editing (and hair pulling) sessions later, **tex.web** made it through **tangle** with no errors.

Eventually I worked up enough nerve to try to compile TEX. **TRY** is the key word in that sentence. I soon discovered that TEX was too large to compile with the Pascal compiler on our system. Both the program and its global variables overflowed their respective limits. All I could think of to do was call Hewlett Packard and ask them if there was some compiler option I was missing. I described my problem to their service representative, who said someone would call me back with an answer. Two days later, I got an answer. They said that the compiler was designed with the theory that if a program grew that large, it would have been developed in sections and compiled seperately. The only thing I could do was to split TEX into pieces and try it that way. The idea didn't exactly thrill me, but I had no other choice.

Splitting TEX into little pieces is not something one should take lightly. It never occured to me that one single program could have over 300 different procedures and functions! I didn't know how small I should make the pieces; should I cut it in half, thirds, quarters? I decided three pieces ought to be enough. It took me nearly two weeks to get all the external definitions straightened out and in the right place. (NOTE: now if I want to split TEX, I let scripts do all the work!) What a mess! Finally I was ready to try again. This time all the pieces compiled and linked together nicely. It even started to run.... started to. I soon found out that it wouldn't read font information, or correctly output **DVI** files. It turned out that this compiler doesn't support packed files of any kind. This meant that special I/O routines would be necessary in order for TEX to communicate properly. After a crash course in integer arithmetic and file storage techniques, I had some I/O methods devised which seemed to work. Now I just had to put them in the change file, re-**tangle** **tex.web**, and then split TEX up again. I can now say with confidence that TEX works on an HP9000.

It was only a day after I got TEX running when I discovered that a whole bunch of other programs, collectively called TEXware, were also on our tape. At first I just left them alone, but then I was told that most of them were necessary in order to test TEX well enough for it to be called TEX! I soon realized that if I had implemented these programs first, TEX would have been much, much easier. Now, all of these programs are running perfectly. My experience with TEX made them quite easy to change.

I would like to express my gratitude to Dr. Norman Naugle, of the Mathematics Department of Texas A&M, and to Tomas G. Rokicki for their assistance on this project. I would also like to mention that their DVI–QMS driver program is included on our tape.

Anyone who is interested in TEX82 for the HP9000 series 500, please contact:

Gregory Marriott or Ken Marsh
Thermodynamics Research Center
Texas A&M University
College Station, TX 77843
(409)845-4940

UNIX TeX SITE REPORT

Richard Furuta

The previous report in the March TUGboat covered through the beginning of February, 1985. This report is being written in early May, 1985.

The most significant developments this time are that the first Unix versions of **METAFONT** were placed on the Unix TeX distribution tape in late March (Version 0.77). The port is being carried out by Paul Richards of the University of Illinois. As noted in last issue's site report, the new **METAFONT** tickled previously undiscovered bugs in the Versatec and Imagen 8/300 device drivers. A fix from Chris Torek of the University of Maryland was installed in late April. At the same time, the device driver directory hierarchy was rearranged to reflect the commonality of the library files used by these two device drivers. Chris also contributed a new filter program, named **dviselect**, that extracts pages from a DVI file to allow them to be printed separately (many of the **dvitype**-based device drivers already perform this function).

Although versions of **METAFONT** are now available for Unix, corresponding font descriptions have not yet been released. Consequently, we continue to provide fonts in the PXL format.

It became increasingly clear to me during this period of time that 300 dot per inch laser printers are becoming rather common. In mid-March, we decided to remove the PXL fonts for the less common 480 dot per inch printers and to use the reclaimed space on the tape to provide a complete set of PXL fonts for the 300 dot per inch printers.

TeX software continues to develop. Version 1.3c of TeX went onto the Unix TeX tape in mid-March. LaTeX also was updated at that time.

In late March, Van Jacobson of Lawrence Berkeley Laboratory contributed **tgrind**, a program that prepares program sources for processing by TeX (a very similar relationship exists between **vgrind** and **troff** on Unix). Subsequently, Van Jacobson and Chris Torek sped the program up, and the faster version went onto the tape in late April. Scott Simpson of TRW contributed a QMS 1200 driver that was placed on the tape in late March. An update in late April added support for the QMS 800.

In closing, I'd like to note that we've received a number of interesting brochures recently from computer companies announcing new 4.2 bsd products. If you port TeX to one of these computers, we're interested in hearing from you. We are also looking for laser printer device drivers for the new printers on the market; particularly the DEC LN03 and the Apple LaserWriter.

VAX/VMS SITE REPORT

Barry Smith

Two items of interest here at Kellerman and Smith

- The new 1.3 release of TeX is available through us, nicely packaged for VAX/VMS V4.0 o later. This includes changes to LaTeX through April, and also the new BIBTEX bibliographic reference manager.
- Macintosh TeX runs, and it runs TRIP.TEX in 6 minutes, 30 seconds real time on the Macintosh XL; one (1) glue setting differs from the reference in the eighth place. We're pleased with the time (a VAX 11/750 runs TRIP in 2 minutes, 40 seconds), especially because we've not put any effort into performance improvement, so there should be room for some reduction in that time. First, though we'd like to see the output — perhaps by the time you read this we'll have a display driver ...