> [TeX] is now very commonly used by scientists, and has changed the look of much that is issued in the area where the concepts of circulation and publication overlap.
>
> Peter Campbell
> "New Looks, New Newspapers"
> *London Review of Books*
> (Volume 10, Number 11,
> 2 June 1988)

TUGBOAT

COMMUNICATIONS OF THE TeX USERS GROUP

EDITOR BARBARA BEETON

VOLUME 9, NUMBER 2 • AUGUST, 1988
PROVIDENCE • RHODE ISLAND • U.S.A.

TUGboat

The communications of the TeX Users Group are published three times per year at Providence, Rhode Island, and are distributed as a benefit of membership to both individual and institutional members.

Submissions to TUGboat are for the most part reproduced with minimal editing, and any questions regarding content or accuracy should be directed to the authors, with an information copy to the Editor.

Submitting Items for Publication

The deadline for submitting items for Vol. 9, No. 3, is September 12, 1988; the issue will be mailed in November.

Manuscripts should be submitted to a member of the TUGboat Editorial Committee. Articles of general interest, those not covered by any of the editorial departments listed, and all items submitted on magnetic media or as camera-ready copy should be addressed to the Production Editor, Alan Wittbecker, at the TUG office.

Contributions in electronic form are encouraged, via electronic mail, on magnetic tape or diskette, or transferred directly to the American Mathematical Society's computer; contributions in the form of camera copy are also accepted. For instructions, write or call Alan Wittbecker at the TUG office.

An address has been set up on the AMS computer for receipt of contributions sent via electronic mail: `TUGboat@Math.AMS.com` on the Internet.

TUGboat Advertising and Mailing Lists

For information about advertising rates or the purchase of TUG mailing lists, write or call Ray Goucher.

Other TUG Publications

TUG is interested in considering for publication manuals or other documentation that might be useful to the TeX community in general. If you have any such items or know of any that you would like considered for publication, contact Ray Goucher at the TUG office.

TUGboat Editorial Committee

Addresses

Note: Unless otherwise specified, network addresses (shown in `typewriter` font) are on the Internet.

TeX Users Group Office
P. O. Box 9506
Providence, RI 02940-9506
or
653 North Main Street
Providence, RI 02904
401-751-7760
`TUG@Seed.AMS.com`

Elizabeth Barnhart
National EDP Dept
TV Guide
100 Matsonford Road
Radnor, PA 19088
215-293-8890

Stephan v. Bechtolsheim
2550 Yeager Rd #3-11
W Lafayette, IN 47906
317-463-0162
`i5f@l.cc.purdue.edu`

Lawrence A. Beck
Grumman Data Systems
R & D, MS D12-237
Woodbury, NY 11797
516-682-8478

Barbara Beeton
American Mathematical Society
P. O. Box 6248
Providence, RI 02940
401-272-9500
`bnb@Seed.AMS.com,`
`bnb@xx.lcs.MIT.Edu,`
`Beeton@Score.Stanford.Edu`

Malcolm Brown
AIR/Systems Development
Sweet Hall 3rd floor
Stanford University
Stanford, CA 94301
415-723-1248
`MBB@jessica.Stanford.Edu`

Lance Carnes
c/o Personal TeX
12 Madrona Avenue
Mill Valley, CA 94941
415-388-8853

S. Bart Childs
Dept of Computer Science
Texas A & M University
College Station, TX 77843-3112
409-845-5470
`bart@cssun.tamu.edu`
Bitnet: `Bart@TAMLSR`

Maria Code
Data Processing Services
1371 Sydney Dr
Sunnyvale, CA 94087
408-735-8006

John M. Crawford
Computing Services Center
College of Business
Ohio State University
Columbus, OH 43210
614-292-1741
`Crawford-J@Ohio-State`
Bitnet: `TS0135@OHSTVMA`

Ester Crisanti
TE CO GRAF snc.
Via Plinio, 11
20129 Milano, Italy

Jackie Damrau
Dept of Math & Statistics
Univ of New Mexico
Albuquerque, NM 87131
505-277-4623
`damrau@dbitch.unm.edu`
Bitnet: `damrau@bootes`

Shane Dunne
Department of Computer Science
Middlesex College
University of Western Ontario
London, Ontario N6A 5B7, Canada
`shane@uwocsd.uwo.cdn`

Allen R. Dyer
2922 Wyman Parkway
Baltimore, MD 21211
301-243-0008 or 243-7283

Maureen Eppstein
Administrative Publications
Stanford University
Encina Hall, Room 200
Stanford, CA 94305
415-725-1717
`as.mve@Forsythe.Stanford.Edu`

Michael J. Ferguson
INRS – Télécommunications
Université du Québec
3 Place du Commerce
Verdun (H3E 1H6), Québec Canada
514-765-7834
CSnet: `mike%tel.inrs.cdn@ubc`

Alberto Formigoni
Dipartimento di Scienze
dell'Informazione
Università degli Studi di Milano
via Viotti, 3/5
20133 Milan, Italy

Jim Fox
Academic Computing Center HG-45
University of Washington
3737 Brooklyn Ave NE
Seattle, WA 98105
206-543-4320
`fox@uwavm.acs.washington.edu`
Bitnet: `fox7632@uwacdc`

David Fuchs
1775 Newell
Palo Alto, CA 94303
415-323-9436

Richard Furuta
Department of Computer Science
Univ of Maryland
College Park, MD 20742
301-454-1461
`furuta@mimsy.umd.edu`

Bernard Gaulle
CIRCE/CNRS
B P 63
91406 Orsay Cedex, France
16.6.941.82.50, ex 3259
EARN: `UCIR001 at FRORS31`

Raymond E. Goucher
TeX Users Group
P. O. Box 9506
Providence, RI 02940-9506
401-751-7760
`reg@Seed.AMS.com`

Dean Guenther
Computer Service Center
Washington State University
Computer Science Building,
Room 2144
Pullman, WA 99164-1220
509-335-0411
Bitnet: `Guenther@WSUVM1`

Doug Henderson
Division of Library Automation
University of California, Berkeley
186 University Hall
Berkeley, CA 94720
415-642-9485
Bitnet: `dlatex@ucbcmsa`

Alan Hoenig
17 Bay Avenue
Huntington, NY 11743
516-385-0736

Don Hosek
Platt Campus Center
Harvey Mudd College
Claremont, CA 91711
Bitnet: `dhosek@hmcvax`

Patrick D. Ion
Mathematical Reviews
416 Fourth Street
P. O. Box 8604
Ann Arbor, MI 48107
313-996-5273
ion@Seed.AMS.com

Helmut Jürgensen
Dept of Computer Science
Univ of Western Ontario
London N6A 5B7, Ontario, Canada
519-661-3560
Bitnet: helmut@uwovax
UUCP: helmut@julian

David Kellerman
Northlake Software
812 SW Washington
Portland, OR 97205
503-228-3383

David Kennedy
Micro Publishing Systems, Inc.
300-1120 Hamilton Street
Vancouver, V6B 2S2 British
Columbia, Canada
604-687-0354

Donald E. Knuth
Department of Computer Science
Stanford University
Stanford, CA 94305
DEK@Sail.Stanford.Edu

Paco La Bruna
TE CO GRAF snc.
Via Plinio, 11
20129 Milano, Italy
2–20 81 50 and 2–27 80 63

Klaus Lichtenwalder
Datapat GmbH
Alter St Georgs Platz
N R 9/8000
München 401
Federal Republic Germany
8913510362

Pierre A. MacKay
Northwest Computer Support Group
University of Washington
Mail Stop DW-10
Seattle, WA 98195
206-543-6259; 545-2386
MacKay@June.CS.Washington.edu

Laurie Mann
Stratus Computer
55 Fairbanks Boulevard
Marlboro, MA 01752
617-460-2610
uucp: harvard!anvil!es!Mann

David Ness
TV Guide
Radnor, PA 19088
215-293-8860

Richard S. Palais
Department of Mathematics
Brandeis University
Waltham, MA 02154
617-647-2667

Mitch Pfeffer
Suite 90
148 Harbor View South
Lawrence, NY 11559
516-239-4110

Arnold Pizer
Department of Mathematics
University of Rochester
Rochester, NY 14627
716-275-4428

Craig Platt
Dept of Math & Astronomy
Machray Hall
Univ of Manitoba
Winnipeg R3T 2N2, Manitoba, Canada
204-474-9832
CSnet: platt@uofm.cc.cdn
Bitnet: platt@uofmcc

Barry Smith
Blue Sky Research
534 SW Third Ave
Portland, OR 97204
800-622-8398; 503-222-9571;
TLX 9102900911
Usenet: tektronix!reed!barry

Rilla Thedford
Intergraph Corporation, MS HQ1006
One Madison Industrial Park
Huntsville, AL 35807
205-772-2440

Georgia K.M. Tobin
The Metafoundry
OCLC Inc., MC 485
6565 Frantz Road
Dublin, OH 43017
614-764-6087

Antonius J. Van Haagen
1818 East Michigan Avenue, Apt 711
Los Angeles, CA 90033
213-269-1220

Glenn L. Vanderburg
Computing Services Center
Texas A & M University
College Station, TX 77843
409-845-8459
Bitnet: X230GV@TAMVM1

Samuel B. Whidden
American Mathematical Society
P. O. Box 6248
Providence, RI 02940
401-272-9500
sbw@Seed.AMS.com

Michael J. Wichura
University of Chicago
Department of Statistics
5734 University Avenue
Chicago, IL 60637-4931
312-702-8329
wichura@galton.UChicago.edu

Alan Wittbecker
TEX Users Group
P. O. Box 9506
Providence, RI 02940-9506
401-751-7760
aew@Math.AMS.com

Warren Wolfe
Department of Mathematics
Royal Roads Military College
FMO Esquimalt
Victoria, B.C., Canada V0S 1B0
604-380-4592
Bitnet: wolfe@rmc

Dominik Wujastyk
Wellcome Institute for the History
of Medicine
183 Euston Road
London NW1 2BP, England
(01) 387-4477
dow@wjh12.harvard.edu

Ken Yap
Dept of Computer Science
University of Rochester
Rochester, NY 14627
Ken@cs.Rochester.edu
Usenet: ..!rochester!ken

Hermann Zapf
Seitersweg 35
D-6100 Darmstadt
Federal Republic Germany

General Delivery

From the President

Bart Childs

When I first started writing this, I was about to say "there is little to report." That is not the case. The TEX community and TUG have been quite busy and there are many things to report.

You were informed in vol. 9, no. 1 that TUG headquarters was moving to a fire station. It has happened! Our separation from AMS is not like a divorce, it is more like sending your child off to grow up. We are appreciative of the support from AMS in getting us started. TUG will still be a customer of AMS in many ways and we expect to continue to have excellent relations. Please bear with us, because moves are never the easiest things to live through.

Dean Guenther has put together an excellent program for our annual meeting in Montréal. It certainly promises to be another great meeting.

One topic we might want to consider at the meeting is scheduling BOFs for some of the "personal" systems (IBM PC, Macintosh, and others), and perhaps making sure there is adequate representation for users of these systems on the Steering Committee.

I hear that Robert M^cGaffey's committee is sending a lot of paper back and forth in their work on driver standards. That is a big task and any progress will be worthwhile.

We are moving forward with a special study group to try to standardize our classes with syllabi, milestones, checklists, prerequisites, and tests. We are also discussing means to try to increase the success rate of offering courses. Too often, we have to cancel classes due to inadequate enrollment. Alan Wittbecker, TUG's TEXnical Director, will be closely involved in these activities.

Some of you have probably noticed that we have new classes of institutional memberships. Organizations with large sets of TEX users can now conveniently and economically involve more of their TEXers. Texas A&M (of course) and Los Alamos National Laboratories have taken advantage of this.

Los Alamos Sets New Membership Record

Ray Goucher

Los Alamos National Lab set a new membership record when it renewed its 1988 Institutional Membership listing **52** individuals!

At TUG's 1985 annual meeting at Stanford University, the Lab sent half a dozen staff members to investigate the feasibility of adopting TEX. In addition to participating in the meeting, several attended the TEX courses which were being offered. In September, Gary Benson called and asked if TUG would do an in-house TEX course for the Lab. Since then, about two dozen courses at all levels of TEX and LATEX have been conducted at the Lab, attended by more than three hundred staff members.

What an undertaking! What an accomplishment!

Congratulations! And **thank you** to all at the Lab who helped make this endeavor such a rousing success, especially Gary Benson, Gary Doolen, David Kratzer, and Pat Vucenic! And a very special thanks to our instructor, Stephan v. Bechtolsheim, who spent so much time commuting between Indiana and New Mexico that he must have given serious consideration to taking up residence in Los Alamos.

Donald Knuth Awarded Franklin Medal

At the Franklin Institute's Medal Day program on April 13, Donald E. Knuth was awarded the Franklin Medal, in recognition of his work in the fields of computer science and typesetting.

The Franklin medal, established in 1914, is the highest honor bestowed by the Committee on Science and the Arts of the Franklin Institute, and is awarded to those involved in physical science and technology who have advanced a knowledge of physical science or its application. The medal was first awarded in 1915 to Thomas Edison. Other medalists include Wilhelm Roentgen, Edwin Land, Albert Einstein, and Marie Curie.

Knuth's citation recognizes him for the development of TEX, and for the series of books *The Art of Computer Programming*. The same series of books was also recognized by an earlier award, the American Mathematical Society's Steele Prize for Expository Writing, presented in 1986.

TEX in a Publishing Environment: A Survey of Production/Commercial Users

Elizabeth M. Barnhart
TV Guide

How the Question of TEX in the Production Environment Started

In August of 1987, members of **TV GUIDE**'s staff presented a paper at the Annual TUG Conference. A discussion of the needs and problems of production users of TEX as opposed to individual academic users followed.

When we first got involved with TEX, in 1985, we discovered that a large percentage of the TEX community consisted of academic users of TEX, but that few production typesetting applications were using TEX. However, it seems that in the last few years more publishing operations are adopting TEX for parts of their typesetting production.

Sample Problems in a Production/Commercial Environment

Although TEX has many positive features, problems were encountered by production users as they experimented with the TEX language. Discussions with other production users at the Conference in Washington led to some of the following points being raised.

The individual academic user is usually involved with a relatively small quantity of output — from a few pages to perhaps several hundred pages. In contrast, production users deal with much larger volumes of output—and on strict production deadlines—posing quite different processing problems.

In the typical academic environment, one person might key in text through a word processor or PC editor and handle the style and output of the text by the insertion of typesetting commands directly into the text. In a publishing environment, the flow of capturing keystrokes and correcting files may be done by many people in several areas and revisions have to be passed around easily. Output is handled by feeding items through predefined typesetting-specification files created by a specialist.

One problem encountered is that TEX was designed for wide columns. When working with narrow-column type for multi-column output, many adjustments have to be made to the `\tolerance` and penalties that control the line-breaking algorithm in TEX. However, it has been pointed out that as you become more familiar with TEX, making these adjustments is relatively easy.

TEX is a paragraph setting composition language; most other composition languages set type line by line. In line-by-line systems, once a line of type has been hyphenated and justified, it is closed and will not be changed; TEX can rework a paragraph completely differently when one word is eliminated. This can present some problems in a production environment, since knowing exactly where a line breaks is often important, for example when laying out multiple items on a page.

Another difficulty discussed is the fact that when TEX produces a `.dvi` file, the fonts involved lose their identity. They are assigned a number in the font table contained in the "postamble" of the `.dvi` file. Some environments need to be able to convert text stored on a database back to the original format, so they must be able to reconstruct the font calls made in the original text. Sending "notes" to the `.dvi` file using the `\special` command can be used to handle this.

One of the biggest problems that most users faced was the complexity of defining page layouts with "output" routines. There can be many different types of output required in a publishing environment since this is all governed by the type designer's specifications for each job processed. Someone with quite a bit of TEX knowledge has to be available to create these varying output routines. Creating large "runaround" text can be complicated, but with some adjustments to how TEX sees paragraphs, this can be handled.

In the publishing world there is also the question of tracking errors, separating "PEs" (Printer's Errors) from "AAs" (Author's Alterations). TEX has no facility to track this sort of thing.

The Point of this Discussion and Survey

The idea of surveying production users to identify their needs and problems was discussed. I was asked to create a questionnaire and circulate it to the production users of TEX. A questionnaire has been developed; it begins on the next page. You can use this questionnaire to let us know if you have encountered similar problems, or others that are unique to your production environment. Please take the time to fill it out and return it by September 1, 1988. Results of this survey will be published in a later issue of *TUGboat*.

Information gathered here could lead to future articles or a section of *TUGboat* devoted to the problems of production TEX users and solutions gathered from the TEX community.

Production/Commercial TEX Users Questionnaire

Company:

Representative's Name:

Street Address:

City, State, Zip:

Phone:

1. What typeset product is the main output of your organization?
   [ ] General Topic Books
   [ ] Technical Books
   [ ] Magazine
   [ ] Journal
   [ ] Labels
   [ ] Directories
   [ ] Forms
   [ ] Newspaper
   [ ] Internal Documents
   [ ] Other: ____________

2. Are you using TEX now for output of any typeset pages?
   [ ] Yes [ ] No

2A. If yes, what percentage of your typeset output is produced by TEX? ____________

2B. If no, are you experimenting with the use of TEX in your production? ____________

3. In what environment are you using TEX, mainframe or micro?
   [ ] Mainframe (Make/model, operating system?)
   ____
   [ ] Micro (Make/model of machine(s)?)
   ____

4. On what type(s) of device(s) are you producing output? Specify make and model of machine.
   [ ] Laser or impact printer
   ____
   [ ] Typesetter
   ____
   [ ] Outside service bureau
   ____

4A. Is your proofing output produced on a different device than camera copy? If yes, have you had problems with font compatibility, and how have you solved them?
   [ ] Yes [ ] No
   ____
   ____
   ____
   ____

5. How do you feel about the level of training required to use TEX for typesetting? Please explain your answer.
   [ ] Less than other systems
   [ ] More than other systems
   [ ] About the same
   ____
   ____
   ____
   ____

6. Do your keyboarders really have to know TEX, or is it "hidden" from them? (Please explain.)
   ____
   ____
   ____

7. Who creates the code for output routines, etc., in your environment? (Explain.)
   [ ] An in-house TEX guru
   [ ] Consultant
   [ ] Production personnel
   [ ] Other
   ____
   ____
   ____
   ____

8. Do you use Plain TEX or a "standard" macro package? Which package(s)?
   [ ] Plain [ ] Macro package
   ____

9. Where and how do you get fonts not delivered with the standard TeX release?

____

____

____

10. Have you used METAFONT at all in your installation? Explain.

____

____

____

11. What have been some of the problems you have encountered trying to develop the use of TeX in your environment?

____

____

____

____

12. How did you find out about TeX?

____

____

____

____

13. What do you feel are TeX's strong points?

____

____

____

____

14. What do you feel are TeX's weak points?

____

____

____

____

15. What would you change about TeX if you could?

____

____

____

____

16. What sources have you used to help with TeX problems? (Please explain.)

[ ] TeX Users Group in Rhode Island
[ ] AMS Offices
[ ] Copies of *TUGboat*
[ ] Knuth's *The TeXbook*
[ ] TeXhax
[ ] TeXMag
[ ] Courses offered in TeX
[ ] Other

____

____

____

16A. If you have contacted the TUG headquarters, were they able to answer your question or solve your TeX problem for you?

[ ] Yes [ ] No (Explain.)

____

____

____

____

17. Can you think of any areas where the TeX Users Group could be of help to you?

____

____

____

____

18. What do you anticipate will be your future involvement with TeX?

____

____

____

____

Please fill out by September 1, 1988, and return to:

Elizabeth Barnhart
National EDP, TV Guide
100 Matsonford Road
Radnor, PA 19088

Editorial Comments

Barbara Beeton

First, I would like to apologize for the many errors that crept into the last issue. Some new production procedures were being instituted, and some of the final checks that should have been made just didn't get done. Through the end of volume 8, production of TUGboat was essentially a one-person job — mine. The TUG office now has a staff TEXnician, Alan Wittbecker, as announced in the last issue, and he will be the TUGboat Production Editor. This means that he will acknowledge incoming items, maintain the administrative records, assist in the TEXing and makeup, and make sure that everything gets to the printer on time. The EDITOR will still be responsible for what does or doesn't get accepted for publication, and will still nag contributors about accuracy, grammar, and other sundry details of that sort.

Some of last issue's errors are amended by errata or addenda in this issue. One omission, TUG's new telephone number, was beyond our control — the phone hadn't been installed until after the copy was delivered to the printer. Though it's listed elsewhere in this issue, here it is again: 401-751-7760. However, we should have gotten the ZIP code for the new address right; it's 02904. Please change it on the last line of page 5.

Several names and addresses of authors were omitted from the last issue. All have been included in this issue's address list.

The most egregious omission was that of Barry Smith's Macintosh site report. It contained two major announcements. The first was Addison-Wesley's withdrawal from the TEX software business, and Kellerman & Smith's assumption of the distribution of *Textures* (as the implementor of *Textures*, Barry was already handling maintenance). The second announcement was that a new version, 1.01, had been released, and that K&S were trying to let all registered users know it was available. Kellerman & Smith have now gone their separate ways. Dave will continue to handle the VAX/VMS work from his new company, Northlake Software, and will be the new VAX/VMS site coordinator; Barry, at Blue Sky Research, will deal with the Macintosh world and *Textures*. Our best wishes to them both.

In order to help keep future submissions to TUGboat from getting lost, a new mail drop has been set up on the Math Society's computer: `TUGboat@Math.AMS.com` on the Internet. Send your articles and your questions there, instead of directly to me. I'll be checking it regularly, and so will Alan, but if either of us is out of town, someone else will be assigned to check it, acknowledge the receipt of messages and inform the senders how long it will be until someone is actually there to read them.

We keep hearing requests that items from TUGboat be made available on-line. A subdirectory `<TeX.TUGboat>` has been created at Score, where it will become part of the standard the TEX distribution, and it will gradually be populated with macros that have been the basis for articles in the Macros and LATEX columns, and with other selected items. As a start, the macros used to produce TUGboat are already there, along with a couple of sample articles. Our current plan is to start with the latest issue, and work backwards. But if you have a favorite macro from an earlier issue that you'd like to see moved higher on the list, please let us know. (You can use the new mailing address ...) The macro for trees, by David Eppstein (**6#1**, pp. 31 ff.) has already been requested. I will be getting in touch with the keepers of additional repositories on Bitnet (`TEX-L`, etc.) and elsewhere, to arrange for inclusion of these files in their collections.

Finally, I'd like to thank all the readers who have had kind words for TUGboat — that's what makes all of this worthwhile.

Software

New Version(s) of TEX and METAFONT

Barbara Beeton

Not long before the printer's deadline, I received a message from Don Knuth — yet another bug, one that affected both TEX and METAFONT. It's been fixed. We're now up to TEX 2.93 and METAFONT 1.5.

During 1987, twelve bugs were found in TEX and two in METAFONT; so far in 1988 the count is six bugs in TEX and two in METAFONT. Some of them have been pretty obscure (the finders constructed samples akin to the `TRIP` test to do their worst), but some of the bug fixes will solve problems that have been bothering users for a long time without their knowing quite what was wrong.

Changes since the end of last year appear as extracts from the files TeX82.BUG and MF84.BUG, in the supplement at the end of this issue.

Changes to TeX

These are the changes that precipitated each new version of TeX beginning with changes made in November 1987. The reward for being the first to find a new bug in TeX now stands at $81.92.

2.6 (November 1987) Added 10sp to width when shipping leaders, to improve the handling of rounding in device drivers.

2.7 (November 1987) Improved the rounding of negative-width characters.

2.8 (December 1987) Removed a loop that occurred when no hyphenation patterns have been loaded and hyphenation is attempted when a `\lccode` is 1.

2.9 (December 1987) Made the assignment of `\csnames` global, so that they don't disappear when the first definition occurs inside a group.

2.91 (April 1988) Fixed a bug that showed up when trying to process `\outer\def\a0{}\a\a`.

2.92 (May 1988) Fixed a problem with `\patterns`. Also fixed bad handling of file names generated with complex macros; in 2.91, typing `\input\romannumeral6` in response to the `**` prompt, and `\end` on the next line, would result in TeX reporting that it has written a transcript file called "`viTEXPUT`".

2.93 (June 1988) Fixed negative halving in memory allocation, which caused a problem just before running out of memory when *mem_min* is negative.

Changes to METAFONT

Here are the changes to METAFONT. The reward for being the first to find a new METAFONT bug is now $20.48.

1.4 (May 1988) A typo which suppressed detection of an error was fixed. The timing of *scan_declared_variable* was also fixed.

1.5 (June 1988) Comparable to the change for TeX 2.93.

Changes to PLAIN.TeX

The following definitions should be corrected in PLAIN.

```
\def\arrowvert{\delimiter"33C000 }
\def\Arrowvert{\delimiter"33D000 }
\def\bracevert{\delimiter"33E000 }
```

TeX implementors' early warning system

Since Knuth has included me in the list of people he informs whenever he makes any changes to TeX, METAFONT, *C&T* or the CM fonts, I have set up a mailing list of all the implementors and distributors I know about, so that changes can be communicated to them as quickly as possible. The details of all the changes mentioned above have been sent out to the list, and acknowledgements were received from most of the addressees, so TeX 2.93 should be generally available soon.

If you are an implementor or distributor of TeX software and believe you should be included on the list, send me your name and address (preferably a valid Internet address) and a short description of what you're doing that makes you eligible.

64-bit TeX

Bart Childs

Most of us have seen one of the little warnings, "`TeX capacity exceeded ...`" at one time or another. When it is memory that has been exceeded, it is probably most often due to a missing brace or similar error. But some macro packages, like Michael Wichura's PiCTeX,Michael Spivak's *Tables to Die For*, and even LaTeX, can encourage this error in legal uses.

While I was working on the port of Cray TeX and reviewing Don Knuth's tapes on *TeX: the Program*, it dawned on me that we can change TeX to use 64-bit words rather than 32-bit words on most systems. We generally use 128k memory as opposed to the former limit of 64k. When we do this, we have noticed an increase in speed of about 7% under AOS/VS.

Lily Barkovic-Mummert has installed these changes on several Mach systems at CMU, IBM PC-RT, SUN 3, and μVAX. These too are probably faster. We will be publishing the details later. If you are interested in obtaining these change files, please contact us.

Porting TEX to C

Klaus Lichtenwalder
Datapat GmbH München

There are already several C versions of TEX available that claim to have passed the trip test, namely CommonTEX, TurboTEX and the version for the Commodore Amiga. It is also said that parts of MicroTEX are written in C. We have also developed a portable version of TEX in C in a one year project, which now has been upgraded to the latest (as far as we know) Version 2.5. Our version of TEX passes the Trip test (naturally, otherwise we wouldn't dare tell you) and has been very easily ported to a number of different machines, as you will see later on.

The translation process

TEX is supposed to be portable by any means (if you get a running version of the BSD pc Pascal compiler). If you look at the TEX distribution sources, you will find that TEX has actually been written in a meta-language called WEB. If you finally get the necessary things up und running and have your TEX Tangled to Pascal source (which is supposed to show up at the end), you probably might stop trying to port TEX. But if you know something about Pascal and aren't worried about editing 1Mbyte files, as we did, just keep on working, or, better, have your Pascal compiler carry on. So, depending on your Pascal compiler, you could have a running version of TEX or, more probably if you are not on a BSD machine, you will start to look at the 1MByte of error messages you just got. So did we, and encountered a problem with our Pascal compiler (probably a cross-compiled version from a 8/16-bit machine) in handling reasonably-sized (definitely *not* large) arrays. A problem had already shown up in compiling Tangle and Weave, two tools you need for handling WEB files. After trying to fix that problem in Pascal without losing too much efficiency, we stopped relying on Pascal and thought about using something else.

Not too far away from Pascal, and maybe a reasonable choice on a UNIX system, we started to think about C. Yet another reason was a C compiler that never showed up with a bug or the like (even the optimizer wasn't buggy, something you may encounter on some 680X0-machine or other). So with this promising background, we took a close look at the some 23,000 lines of source code and stopped working on that problem soon after, because vi didn't seem well suited for this problem, since standard vi is limited to a 256K input file! Handling multiple files is not comfortable and what about viewing two files at the same time? We split the source into many smaller units, so as to drastically reduce compile time when making minor changes in just one procedure, and to have better response time in handling text units of a few K instead of a few hundreds. People invented a make facility, so why not use it? (God bless UNIX!)

At this time, we got the source of a fairly big version of EMACS and tried to port it to our machine (then a V.0 68000 machine). This decision turned out to have a major impact on the successful port of TEX, because of EMACS' sophisticated text handling commands and not easily surpassed size restrictions, and last, but by far not least, the modes that help you edit programming languages.

But anyway, having EMACS doesn't solve porting problems. First thing then was to invent some kind of Pascal beautifier. The rewrite process started afterwards. Line by line the Pascal source was replaced by the equivalent C code. There EMACS was of great help with global replace functions over all of its buffers, and with its keyboard macros. The latter were especially useful for transforming Pascal control structures to their C equivalents. We didn't intend to transform TEX into a hell of a C program (who would dare to change Knuth's intentions, or, worse, algorithms?), but to simulate Pascal restrictions as closely as possible so as not to lose portability (whatever degree of portability you expect). In the middle of the rewriting process (and after buying "TEX: The Program") we learned of some useful macros and `defines` we re-introduced into the Tangled source code for flexibility. So this prolonged step of rewriting into C took about half a year. The process of convincing the C compiler that the stuff he was reading was actually C did not take too long. There were quite a lot of misspellings and misconceptions and the like to get rid of, if you expected the program to do more than just print out the banner message "`This is TeX...`".

The major problem

One misconception, however, was of particular importance to the portability of the C version that slowly came into existence. In the WEB versions there are provisions for machines that do sign extensions, and for machines that don't. We ignored the preconditions, and, as always with a 50% chance, you get the wrong half. When we learned about the problem and also that this source wasn't intended for our machine, we went over the source code once again and spotted it with a handful of type casts.

The net effect is, if your C compiler knows how to handle casts (up to now, every C compiler we could test did), that this source runs on both types of machines.

Passing the Trip test

The important milestone after the banner is the Trip test. Needless to say, it didn't run at once. In fact, there were some very subtle bugs introduced while rewriting. It took another half a year to succeed with this test. One of the main problems lay in the input routine, where we didn't use Knuth's raw version, but the optimized version that happened to be in the change file (ever heard of a change file?). Needless to say, the algorithms and data structures in TeX are computer proof; that means, that if you have a compiler that deserves this name, you get this program running.

Sure enough, people learned about our project and asked for a port, if we ever got it running. Most of the time these people were more optimistic about a possible success than we were. But then we could make the (ultimate) test for portability.

The one thing we learned is, that TeX (whether in Pascal or in C or whatever) is not only a typesetting system, but also a compiler test system. There were some problems compilers introduced with the input of TeX, but if you wanted to demonstrate the bug to the computer or compiler distributor, you couldn't reproduce the error with a normal sized program. We encountered the fact that fixed array locations like `mem[32760]`, as happen to be used as kind of register in the typesetting processor TeX, will be translated to anything, but definitely not to the locations you would expect. Also you have to cope with the most tricky optimizers, which try to keep the program small enough by optimizing procedures away, or deleting the index in `z = mem[z]` before using it. But these problems were not too often encountered, and after tuning some I/O statements not for efficiency but for portability, we now have the following ports:

Cromemco V.0 and V.2
PCS Cadmus 32Bit UNIX Systems
ALTOS UNIX and XENIX Systems
Convex with 4.2 UNIX
AT&T 3b2 running V.0 and V.2/V.3
HP Series 9500 and 93XX under HP-UX
IBM RT under AIX

The only preconditions we pose are that we have a true 32-bit CPU (not an 80286) and we prefer UNIX or UNIX look-alikes, but we don't insist on this (as people insist on a VMS version).

Extensions

With this TeXinC version we started a cooperation with a German typesetter. In this project we designed an extended TeX program, which we call PhotoTeX, to cope with the possibilities available with phototypesetting machines, and made some adaptations for German respectively European environments. The PhotoTeX Program understands two additional keywords, `setsize` and `slantsize`, so that we are capable of handling dynamic fonts in the typesetting machine. Also we had to create metric files (tfm-files) for the fonts that are resident in phototypesetting machines, as there are machines that are not able to download fonts. Another hard problem was to find the right kernings, as typesetters need them.

An additional problem for the German environment is hyphenation of words with Umlaute (and other special characters you encounter in a European environment). At the moment, there are two solutions we know for hyphenation, both coming from the University in Bonn, Germany. One is to fool the hyphenation routine, while the other, and by far better, solution requires a minor change in the METAFONT description, recreating the fonts, and an addition in the dvi driver. We preferred the second approach for our German version of TeXinC.

TeX Adapted to CWEB

David Kennedy
Micro Publishing Systems, Inc.

This article announces TeX in CWEB, a new starting point for TeX ports. We have recently completed the translation of TeX to CWEB, a version of Don Knuth's WEB system of structured documentation, entirely rewritten in C, with many changes to take advantage of features found in C, but not in Pascal. (For a more complete description of CWEB refer to the TUGboat article: *WEB Adapted to C, Another Approach* by Silvio Levy, April, 1987).

Although this is a commercial venture, and the TeX translation is proprietary, we are offering a copy of the binary and/or source code for a reasonable license fee. We are also planning a fall 1988 commercial release of our fully TRIP-certified version of TeX for the PC and plan to release UNIX

System V, XENIX and OS/2 versions in the near future.

Given the current availability of TEX implementations on microcomputers, a TEX site needs to acquire a different implementation for each of its systems, assuming that a version exists. Our objective is to provide an implementation of TEX which is easy to port to various microcomputers and operating systems and preserve the documentation features of CWEB so that a system administrator can maintain one source to TEX for various small systems.

The original development work was done in MS-DOS on an AT compatible computer using the Microsoft C Compiler version 5.0. We are interested in hearing from institutions and individuals who would like to port TEX in CWEB to other unique operating systems, computers, and/or languages.

Adapting TEX to CWEB provides simultaneously a language like TEX for formatting and a language like C for programming. Once the WEB is translated to CWEB, a program called TANGLE is used to produce the C source code. Another program called WEAVE is used to produce a structured document, so that the large system can be understood entirely in terms of small modules and their local interrelationships. Any changes such as splitting the C source into multiple files is handled automatically within TANGLE (the entire C source produces an object file greater than the 64K limit for the Microsoft C Compiler). Any system specific changes such as those necessary to accomodate the 64K byte address space limitations imposed by the segmented architecture of the 8086 processor can be made in the change file included by TANGLE.

Now that the original translation of the TEX WEB to CWEB has been completed, and the major port for MS-DOS has been done, ports to other C environments will follow quickly. The most important thing is maintaining the source to TEX in a generic form while allowing for the smooth integration of system-dependent changes.

Fonts

Some Useful Variations of Standard Fonts

Glenn L. Vanderburg
Texas A&M University

The old "Almost Computer Modern" font set had a font called amssmc40. It was used for the chapter titles in the first edition of *The TEXbook*. It also turned out to be very popular for making signs. When the Computer Modern fonts were released, there was no equivalent font in the standard set. It was replaced by cmssdc10 scaled 4000. This is essentially the same font, with some minor improvements and one major disadvantage: the unusual magnification. Unfortunately, many DVI drivers have difficulty dealing with fonts which are not magnified by one of the standard "magstep" quantities.

To remedy this, I added the following line to cmssdc10.mf just after input cmbase:

```
numeric Pt#; 1Pt#=4pt#;
```

and changed all occurrences of 'pt#' in the rest of the file to 'Pt#'. I called the new file cmssdc40.mf. The font produced by it is not exactly equivalent to cmssdc10 scaled 4000, but the differences are negligible.

A similar modification to cminch.mf yields another extremely useful font. I noticed that a couple of signs I made with cminch had a harsh tone because of the capital letters, even when the wording was very friendly. Changing the 'generate title;' line at the end of cminch.mf to 'generate roman;' creates a cminch font with lowercase characters (among other things). It takes up a lot more disk space, but many people use it here at Texas A&M. It should not be called cminch, because it's not the same font. I call it cmssbx104 (when the capital letters are one inch high, the font size is 104 points). This will cause problems for systems (like CMS) which have severe restrictions on filename length, but I have been assured by "people who know" that there is a convention for handling this problem.

Modifications like this are extremely simple to make, and can be quite useful. A new version of cminch with characters one centimeter high would be nice; it could be called cmcm. (Along the same lines, Tom Reid once observed that cminch scaled\magstep1 might reasonably be called 'cmdecifoot'.) I've seen several people requesting cmssdc40; it turns out to be very easy to create, and I think people will enjoy using it.

FONT FORUM

Georgia K.M. Tobin

Designing for Low-Res Devices

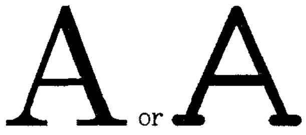
As I have pointed out on more than one occasion in this column, the great strength of METAFONT as a font design tool is its potential for versatility. With a view to maximizing that potential, I decided to see just how much versatility I could get by *never* putting anything as low-level as a **draw** or a **penstroke** in my code for Liber, the Century Schoolbook-inspired font that I am currently working on. My reasoning was, since letter routines are at the lowest level in the design hierarchy, they tend to multiply the most quickly. (This notion was codified as Georgia's Empirical Observation #32: Modifying 128 routines every time you need a new font style gets real annoying real fast.) Modifying a smaller number of routines shared by those 128 letters, while distinctly harder in the short term, should (I reasoned) save effort and contribute to design coherence in the long term.

To wit, the following describes the letter A:

```
beginchar(65,cap#,0); "ms/uc/a.mf";
AssureSymmetry(2);
penpos1(ucHairP,0);
penpos2(ucHairP,0);
penpos4(ucStemP,0);
y1=h;  x1=1/2[LftEdge,RtEdge];
x2=LftEdge+.35HairSerWd;  x4=RtEdge-.4HSerWd;
y2=y4=0;
Crotch(1,2,4,uc,serif);
z4r-z4l=z1r-z.dummy;
PlaceEdges(5,1,4);  PlaceEdges(6,1,2);
y5r=y6r=y5l+ucHairP;  y5l=y6l=1/3h;
Stroke(5,6);
endchar;
```

In addition to three "housekeeping" or calculation macros, I call two macros that are responsible for actually depicting character parts. Crotch, the more complicated of the two, handles the apex and legs of the character, and the serifs that appear at the bottom of the legs. Stroke simply adds the crossbar.

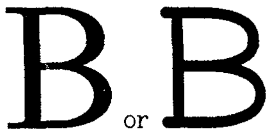
Now, depending largely on the way in which the macros are defined, we get



Similarly, the code for B:

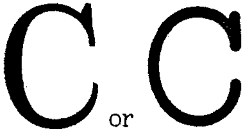
```
beginchar(66,cap#,0); "ms/uc/b.mf";
penpos1(ucStemP,0);  penpos2(ucStemP,0);
penpos3(ucHairP,90);  penpos6(ucHairP,90);
penpos4(ucFatP,0);  penpos7(ucFatP,0);
penpos5(ucHairP,270);  penpos8(ucHairP,270);
y1=y3r=h;
x1=x2=LftEdge+.5HSerWd;
y2=y8r=0;
x3=x5=x1r;
y5=y6=37/72h;
x4r=RtEdge-Slab;
x6=x8=x1r;
x7r=RtEdge;
StemStroke(1,2,uc);
LoneSer(BotSer,LftSer,2,ucStemP);
LoneSer(TopSer,LftSer,1,ucStemP);
ModBowl(3,4,5);  ModBowl(6,7,8);
endchar;
```

can yield



and the code for C,

```
beginchar(67,cap#,0); "ms/uc/c.mf";
penpos1(ucHairP,90);
penpos2(ucFatP,180);
x1r=31/57[LftEdge,RtEdge]; y1r=round h+vo;
x2r=round LftEdge; y2=h/2;
x3=x1; y3l=0-vo;
x5r=RtEdge-1/2Slab; y5r=48/72h;
z6r=z1r;
x4l=RtEdge; y4l=25/72h;
DemiBowl(1,2);
QtrBowl(3,4,uc);
QtrBowlSer(6,5,uc);
endchar;
```



At first blush, the representation of the letters on the right might seem a giant step backward. I have forsaken the bracketed serifs of the top example for simple slab serifs. I have quite forgotten the distinction between hairline and stem weights to use a monoweight pen for both. If you could read the actual code for the macros, you'd see that I have discarded the use of **penstroke** to render the image, and have used the **draw** command instead to create the image on the right.

Why I might be interested in such an apparently simplistic approach to depicting characters can be better seen by comparing the actual pixel images of the top and bottom styles shown above, when generated for a ten point font for an 80dpi device:

```
    **
    **
    **
   *  *
   ****
  *   *
  *    *
X***  ***
 ****
  *  *
  *  *
  ****
  *   *
  *   *
  *  **
X*****
  ****
 **  **
 *    *
 *
 *
 *    *
 **   *
X ****
```

```
     *
     *
    * *
    * *
   *****
   *   *
  *     *
X***   ***
 ****
  *  *
  *  *
  ****
  *   *
  *   *
  *   *
X*****
   ***
  *   *
 *    *
 *
 *
 *    *
  *   *
X  ***
```

and for ten point for 118dpi:

```
      *
     **
     * *
     * *
    *  *
    *   *
    *   *
   ******
   *    *
  *      *
  *      *
X***   *****
 *******
   *   **
   *    *
   *    *
   *   **
   *****
   *    *
   *     *
   *     *
   *     *
   *    *
X*******
   *****
  *    *
        *
 *      *
 *
 *
 *
 *
 *      *
        *
  *    *
X  ****
```

```
      *
      *
     * *
     * *
    *   *
    *   *
    *   *
   *******
   *     *
  *       *
  *       *
X***     ***
 ******
  *    *
  *     *
  *     *
  *    *
  ******
  *     *
  *      *
  *      *
  *      *
  *     *
X*******
    ***
   *   **
  *     *
  *     *
 *
 *
 *
 *      *
  *     *
  *     *
   *   *
X   ***
```

In a nutshell, these apparently more simplistic techniques – monoweight, drawn strokes with slab serifs – go a long way towards creating satisfactory images for VCR (very coarse raster) devices.

It is curious that it was in attempting to automatically produce these almost crude-looking versions of characters that I needed to make the overall design approach much more sophisticated. In addition to telling METAFONT how a character will be created using VCR code, I also have to tell METAFONT how to decide when to use that code. That is an area in which I am still experimenting. It's clear enough that I want the VCR macros to be used when I am rendering a letter for 10 point on a 118 dpi screen, and that I can go with the standard RDR (reasonably decent raster) macros for a 24 point face for the same device; but where is the demarcation between VCR and RDR to be drawn? At present, I am achieving the results I want with a short but inelegant series of **if** tests. Clearly, I need to come up with an algorithm, which very probably need not be much more complicated than determining the number of pixels in an em; if less than a certain magic number, METAFONT is to use VCR definitions rather than RDR ones.

Once METAFONT "knows" it is dealing with a VCR character (in my current approach), it proceeds setting a toggle and testing with it to load files containing the appropriate definitions.

Take for instance the subroutine we mentioned earlier, Stroke, which renders the crossbar on the 'A'. When we are doing an RDR version, we use this definition of Stroke:

```
def Stroke(suffix $, $$)=
    penstroke z$e--z$$e;
enddef;
```

when it's VCR style we're after, we use this one:

```
def Stroke(suffix $, $$)=
    pickup ucPen;
    draw z$--z$$;
enddef;
```

The first appears in a file called mstxt_RDRrtns.mf, the other in mstxt_VCRrtns.mf. A controller file that handles loading of all requisite code at run time tests on the value of the boolean VCR and inputs one or the other, as necessary. In fact, I could as well have accomplished the same thing by having a file shared by VCR and RDR fonts; the common definition of Stroke would read:

```
def Stroke(suffix $, $$)=
if VCR:
  pickup lcPen;
  draw z$--z$$$;
else:
  penstroke z$e--z$$e;
fi;
enddef;
```

Naturally, deciding *what* definitions to input is the least of our worries. Writing code that will perform robustly under such adverse conditions as a coarse raster is tricky. A quick scan of the leftmost pixel representations above hints at the problems encountered. I haven't been able to choose between blotchiness (two pixels where one is needed) or discontinuity (no pixels where one is needed) as the single most irksome problem; both are shown in 'B' and 'C'. 'A' shows another bugaboo, character symmetry. This is by no means exclusive to VCR images; it's just that the low resolution constraints have a way of keeping code honest. Quite simply, a one pixel shift off-center of a V or T will not announce itself nearly as emphatically on a grid of hundreds of pixels wide as on one under ten pixels wide. I have written a routine AssureSymmetry which modifies the *apparent* width of a character as needed for symmetry while leaving the true (tfm) width untouched.

Moreover, as I hinted at the outset of this article, there are other factors besides the macro definitions themselves which contribute to a good VCR design. A glance at the sample characters will suggest a few such factors: pen weights, character parts such as slabs and serifs, and even more subtle things such as the amount of "roundness" in an arc. As with the macro definitions, I segregate these into VCR- and RDR-specific files, and input the appropriate ones at run time.

The results I have obtained so far by using these techniques have been very encouraging. The face itself is something of a x-height hog (i.e. the x-height is large in proportion to point size), and this predisposes it to be an excellent screen font. This predisposition, combined with the elimination of blotchiness, discontinuity and asymmetries, yields 118 and 80 dpi screen text that is attractive as well as clear at ten point; and easily legible down to 5 point. (Readers will find chapter 24 of *The METAFONTbook* quite illuminating in this regard.)

In addition to resolving some of the theoretical questions I mentioned above, some more mundane work needs to be done. I need to code VCR routines for bold style text, as well as for the math fonts. Italic and slanted fonts, it seems, tend to suffer greatly when adapted to VCR devices; I hope to come up with a scheme that reduces some of the unsightly jagginess. I expect that adapting the rotated pens that draw my upper case math symbols script to VCR style will prove easier than dealing with the italic text.

ABCDEFGHI
JKLMNOPQRS
TUVWXYZ

I hope this brief summary of my hierarchical approach to a system of fonts coupled with a choice of character description techniques tailored to the target output device has demonstrated to the reader the marvelous dichotomy of good METAFONT design: how METAFONT has, at once, the *flexibility* to produce custom bit maps and the *consistency* to maintain the same metrics information for all bit maps.

Have some fun with METAFONT, and have a hi-res day!

G.K.M. Tobin 19 May 88

This document originally printed at 300 dpi with GT Liber fonts.

TeX Fonts and Suggested Magnifications

Bart Childs

Several tables follow that indicate my suggested grouping of the many fonts that are available in most TeX systems. These fonts are all in the cm family developed by Donald E. Knuth or are in some manner related to them. The data given for each font is a maximum magnification required or suggested. Blank columns are included so that all tables have the same layout.

In some of the tables I will indicate as different fonts the same typeface at different sizes. The numeric values in the data columns will indicate the maximum TeX magnification specified in `plain.TeX`, LaTeX, SliTeX, and the distribution from Texas A & M for Data General systems. Numeric entries followed by a "p" indicate the magnification of `plain.TeX`'s *preloaded* fonts.

Table 1 contains the sixteen fonts named in `plain.TeX`. Most fonts are furnished in *default* magnifications of 0, half (h), 1, and 2. Six of the sixteen are furnished in larger magnifications as well. These (except for `cmex10`) are intended for use in titles. Two additional fonts are also furnished in large magnifications for titles. These are the sans serif fonts, `cmss10` (Table 2) and `cmss17` (Table 4).

In Table 1 it is shown that LaTeX uses `cmr10` scaled to `\magstep5`. I think it would have been more consistent to have made the larger fonts from lesser scaling of `cmr17`. Indeed, `cmr12` and `cmr17` are used in LaTeX only without magnification. See the last section of this article for more discussion on font design sizes.

Table 1. The Big Sixteen

font	TeX	LA	SLI	DG
cmbx5	0			2
cmbx7	0	0		2
cmbx10	0	3		8
cmex10	0	0	3	3
cmmi5,7	0	0		2
cmmi10	0	4		2
cmr5	0	0		2
cmr7	0,4p	0		2
cmr10	0	5		8
cmsl10	0	h		7
cmsy5,7	0	0		2
cmsy10	0	4		2
cmti10	0	h		7
cmtt10	0,2p	h		7

The fonts in Table 2 are required by LaTeX. These are all the fonts required by LaTeX except for those already indicated in Table 1. (I tried to make sure that each font appeared in only one table.) The scatter of magnifications at first seems weird, but they are appropriate. This data is based on the files `lfonts.TeX` and `sfonts.TeX`.

In `lfonts.TeX` and `sfonts.TeX`, many more fonts are specified in lines that are commented out. You can access these additional fonts by removing the `%` from the beginning of the relevant lines; however, these fonts are not shown in the table, and may or may not be included in the distribution.

Table 2 LaTeX Fonts

font	TeX	LA	SLI	DG
cmbx9	0p	0		2
cmbx12		0		2
cmmi6,9	0p	0		2
cmmi8	0p	0	8	8
cmmi12		0		2
cmr6,8,9	0p	0		2
cmr12,17		0		2
cmsl12		0		2
cmss10		h		8
cmss12		0	1	2
cmsy6		0		2
cmsy8		0	8	8
cmsy9		0		2
cmti8,9	0p	0		2
cmti7,12		0		2
cmtt9	0p	0		2
cmtt12		0		2
circle10		0	0	2
circlew10		0	0	2
lasy5,6,7		0		2
lasy8		0	8	8
lasy9		0		2
lasy10		4		4
line10		0	0	2
linew10		0	0	2

The fonts in Table 3 are used only in SliTeX. Many of us have attempted to be economical by modifying these by taking into account that scaled fonts could be replaced by larger point size fonts. For example, `cmss17 scaled \magstep2` could be used rather than `cmss10 scaled \magstep5`. *However, the slides just won't look as good!* Thanks to Barbara Beeton for pointing this out to me.

You may also have noticed that the `lcmss*8` fonts have rather large magnifications. The smaller magnifications are not referenced anywhere.

Table 3 SliTeX Fonts

font	TeX	LA	SLI	DG
lcmss8			3..9	9
lcmssb8			3..8	8
lcmssi8			3..8	8
cmtt8	0p		3..8	8

The fonts in Table 4 are more often used for emphasis than for text in documents that I have seen. These fonts should be considered optional in most installations. Some of these fonts could be provided at larger magnifications to complement the title fonts shown in Table 1. The font `cmss17` is listed here and is furnished in large magnifications for use in titles. Glenn Vanderburg's `cmssdc40` and `cmincha` are included as well. (Please see his article on page 125.)

Table 4 Fonts for Emphasis

font	TeX	LA	SLI	DG
cmb10				2
cmcsc10				2
cmdunh10				2
cminch				2
cmincha				2
cmss17				8
cmssbx10	2p			2
cmssdc10				2
cmssdc40				2
cmssi10				2
cmssq8	0p			2
cmssqi8	0p			2
cmtcsc10				2
cmtex8,10				2
cmtt8				2
cmu10				2
cmvtt10				2

The fonts in Table 5 are definite candidates for saving disk space. They are mainly common fonts at different sizes that are not commonly used. The `cmff` and `cmfib` fonts are not related, other than being "cm". One is a "funny font" and the other is based on Fibonacci numbers.

Table 5 Some More Fonts

font	TeX	LA	SLI	DG
cmbx6,8	0p			2
cmbxsl10				2
cmbxti10				2
cmff10				2
cmfib8				2
cmit10				2
cmmib10				2
cmsl8,9	0p			2
cmsltt10				2
cmss8,9				2
cmssi8,9				2
cmtex9				2
lasyb5..10				0

Most users will probably not need the fonts in Table 6 unless they are playing around, using METAFONT, or creating documents about TeX and METAFONT. The font `manual` is sometimes referred to as `manfnt`. It is suggested that you make a copy and name it such; in the world of AOS/VS, we simply create it as a link.

The `logo10` and `manual` fonts are included at the magnification indicated because I give lots of talks about TeX and its components.

Table 6 Special Fonts

font	TeX	LA	SLI	DG
flogo				0
gray				0
logo8				0
logo9				0
logo10				4
sklogo				0
manual				4

The AMS fonts are also included in the DG distribution, though not in the tables.

Finally, you might notice that I don't show any invisible fonts. I explained how we handle these in TUGboat **8**#3.

On font design sizes

Pierre MacKay has argued eloquently that using fonts designed for particular sizes improves the quality of a document's appearance. As TUG Site Coordinator for Unix-flavored TeX, he has provided a number of tools in the Unix distribution to provide TeX users the ability to decide for themselves what fonts to use. His opinion is clear in this quote from TeXhax 1988, #51:

> No less a person than Brian Reid has announced in an interview in the *Unix Review* that no one cares about distinctive design sizes any more, and that one 10 or 12 point master will do for all. I disagree, and it seems that a lot of TeX and LaTeX users disagree. The undesirable aspects of using one design for all sizes are even more noticeable in 5, 6 and 7 point fonts than they are in 17, 20 and 25 point fonts, but they are noticeable at both ends. METAFONT makes it unnecessary to coarsen font designs in this way, and I look for it to infiltrate the printing world with considerations of quality just as TeX has.

The Many Faces of TEX A Survey of Digital METAfonts

Dominik Wujastyk

Contents

1 Introduction

This article seeks to give a reasonably complete survey of the fonts and METAfonts that are currently available for use with TEX. Although I have been primarily interested in cataloguing fonts designed with METAFONT, I also wanted to include information about any other fonts that have been successfully used with TEX, i.e., fonts with TEX Font Metric files, and a mechanism for the creation of the appropriate ligatures and kerning pairs, be it within the TFM file, or by means of a preprocessor.

I first started compiling this article late in 1987, as a note to myself and my immediate Indological colleagues. But it seemed little extra work to include more information in it about other fonts that I had heard of, and doing this greatly widened its usefulness to TEX users in general. But you may still detect a slight Indic leaning.

For those with access to the academic computer networks, I post information about revisions of this article to TEXhax from time to time, and send the latest version of the article to `score.stanford.edu`, where it ends up as `WUJASTYK.TXH` in the directory `TEX.TEXHAX`. This way, people will know from TEXhax whether there is enough new stuff for it to be worth downloading the whole article afresh.

I would be grateful for any relevant information that is not already mentioned and, of course, for any corrections. While keeping the memo reasonably concise, I have given all the useful information that I currently have. I have also given everything I know about how to get more information about each font, so follow those leads rather than contacting me directly, in the first instance.

This article is made up of information given to me by others, both in person and through general publication in *TUGboat*, TEXhax and the net. My sincere thanks to all the contributors.

2 Computer Modern

It may seem odd to start with Computer Modern (CM), the typeface family that most TEX users use most of the time, since it was created side by side with TEX, and is included in all distributions of TEX. Nevertheless, I feel that there is an important point to be made about CM.

When Knuth developed METAFONT, one of the central ideas of the whole project was that of producing *parametrized* typefaces. In Douglas Hofstadter's memorable phrase, METAFONT is a 'knobbed category machine'[1], and when Knuth announced the capabilities of METAFONT to the world in *Visible Language* it was precisely this parametrization that he emphasized and demonstrated so brilliantly.[2] Later on, Knuth explained that the whole inspiration for METAFONT had arisen from the three ideas of *pens, parameters* and *programs*.[3] Hofstadter argued, wrongly I think, that this idea contained basic flaws, and that some of Knuth's implications about using METAFONT to generate different typefaces by twiddling the 'knobs' of a single underlying typeface

[1] Douglas R. Hofstadter, 'Metafont, Metamathematics and Metaphysics: comments on Donald Knuth's Article "The Concept of a Meta-Font" ', *Visible Language* **16** (1982), 309–338. This article, and selections from the discussion which it engendered in the pages of *Visible Language* (henceforth *VL*), were republished as chapter 13 of Hofstadter's *Metamagical Themas* (New York, 1986^2).

[2] 'The Concept of a Meta-Font', *VL* **16** (1982), 3–27.

[3] Donald E. Knuth, 'Lessons Learned from Metafont', *VL* **19** (1985), 35–53.

description were misleading.[4] Knuth made it clear that he had never meant to imply that all typefaces could usefully be combined into one single METAfont (although he did not actually deny the feasibility of such an endeavor), and again emphasized the desirability of trying to incorporate variability into a design.[5]

In view of the vigor with which Knuth has maintained the idea of parametrization, both in discussion and in the actual implementation of Computer Modern, I am very surprised that the following survey of TEX fonts does not include a single example of a new typeface created from CM by changing its parameters. In Knuth's own hands, CM is utterly plastic, as was demonstrated so startlingly in his article 'The Concept of a Meta-Font' referred to above, and by the inclusion of such fonts as Computer Modern Funny Roman (CMFF), Unslanted Text Italic (CMU), and the delightful CMFIB (which uses the Fibonacci series for the ratios of several of the CM parameters) in the standard distributions of CM. But none of us has taken up the challenge, implicit in the 62 parameters of CM, to produce a new face for general distribution. It would be very nice, for example, to produce a full set of CMFIB, with bold, slanted, italic, typewriter and other versions. And someone should pick up the gauntlet thrown down by Hofstadter, and try to produce a Times Roman, or a Baskerville or some other familiar face from the CM programs. Knuth has said that we should not blindly copy the old masters, without trying to understand why they produced what they did.[6] How interesting it might be, then, to try to manipulate the parameters of CM to produce a different, but recognizable family of faces. And if the experiment failed, the reasons why it did so would themselves be of great interest. The first sentences of the Introduction to *Computer Modern Typefaces* are:

> Infinitely many alphabets can be generated by the programs in this book. All you have to do is assign values to 62 parameters and fire up the METAFONT system; then presto — out comes a new font of type.

Let's do it, but of course in the best possible taste!

3 Devanāgarī

Devanāgarī is the alphabet used for writing and printing Sanskrit, Hindi and several other languages of South Asia, both ancient and modern.

3.1 Knuth

As far as I know, Donald Knuth coded the first Devanāgarī character to be created with METAFONT. This was the single syllable *la*, which Matthew Carter gave to Knuth in 1980 as a challenge to test the capabilities of the then nascent METAFONT. The smoke proof of the character, and several interesting remarks about the experience, were published as 'My First Experience with Indian Scripts', in *CALTIS-84: Special Issue on Calligraphy, Lettering & Typography of Indian Scripts*, (Proceedings of a Delhi 1984 conference).

3.2 Ghosh

An early Devanāgarī font was designed with old METAFONT (MF-in-SAIL) by P. K. Ghosh during a visit to Stanford in 1982–83. Ghosh published what he had done as Stanford Computer Science Report 965: *An Approach to Type Design and Text Composition in Indian Scripts* (Stanford, 1984). One of the valuable aspects of this work was that Ghosh worked from Devanāgarī characters designed and drawn for him by the famous Bombay calligrapher R. K. Joshi. Drawings of these, on a grid, are published in the *Report*. Unfortunately, Ghosh's work was done in a now superseded version of METAFONT, and was not fully worked out at the keyboard level. It also lacked a number of the conjunct consonant clusters necessary for fine Indian typography. The report, however, remains of considerable interest for general background. The source code is available at the University of Washington, through Pierre MacKay (address below), and presumably at Stanford (try Emma Pease). Ghosh has said explicitly that he has no objection to others doing further work on it.

Contact

If you wish to contact Ghosh he can be reached at the following address:
National Centre for Software Technology,
Gulmohar Cross Road 9,
Juhu, Bombay 400 049,
India.

3.3 Velthuis

The only fully worked out version of Devanāgarī presently available is that of Frans Velthuis.

[4] I agree with the refutation by Geoffrey Sampson, 'Is Roman Type an Open-Ended System: A Response to Douglas Hofstadter' *VL* **17** (1983), 410–412, in spite of Hofstadter's reply in pages 413–416 of the same issue.

[5] *VL* **17** (1983), 417.

[6] *VL* **17** (1983), 417.

The Font

In November 1987, Frans Velthuis completed version 1.0 of a Devanāgarī METAfont for TEX. He has written METAFONT code for all the *akṣaras* (syllabic characters) necessary for Hindi, and most of those for Sanskrit too, although in the latter case some *virāmas* are used. Frans intends to produce a special Sanskrit version of his font in the future. Also included are the Devanāgarī numerals, *anusvāra*, *virāma*, *daṇḍa*, *candrabindu*, *visarga*, *avagraha*, full stop, and the superscript abbreviation circle.

Usage

You prepare your TEX or LATEX file normally, and mark any Hindi portions, typed in a simple Roman transliteration, with a font marker, thus: `{\dn ...}`. At the top of the TEX file you `\input` a file called `DNMACS`; in LATEX, a `DEV.STY` file is provided which inputs the necessary macros, and automatically makes appropriate font size changes. Frans provides a preprocessor, `DEVNAG`, available compiled for several systems, or in Pascal or C, which reads your file and converts the Hindi transliteration into the appropriate codes for Frans's font. The converted file is then processed by TEX or LATEX in the normal way, and the resulting `DVI` file can be printed using a standard `DVI` output program. The portions of Hindi text originally in Roman transliteration will be printed in Devanāgarī, with full use of conjunct consonants (*sandhyakṣaras*), etc.

Quality

The quality of the typeface is excellent, with full calligraphic molding of the curves and loops, like some of the best handwriting of manuscript scribes using a broad nib.

Terms of Availability

Frans will sell a set of four or five sizes of the Devanāgarī fonts, at the printer resolution you specify (Epson-type 9 pin matrix, 24 pin matrix (180 × 180, 360 × 360, 180 × 360), write-white laser, or write-black laser), together with the compiled code (specify VAX/VMS, SUN, Cyber, IBM/PC, Atari ST) of the text preprocessor DEVNAG, for $119. The METAFONT source programs are not at present being made generally available.

Contact

Frans J. Velthuis,
Nyensteinheerd 267,
9736 TV Groningen,
The Netherlands.
Bitnet: `velthuis@hgrrug5`

Further information

A note about Velthuis's Devanāgarī font appeared in TEXhax, 1987, issue 93. Velthuis intends to publish a full account of his font in a future issue of *TUGboat.*

4 Tamil

4.1 Arthanari

According to Emma Pease (network response on 10 November 1987 to my query in TEXhax 1987, issue 93) a basic set of Tamil characters for TEX was designed and created by T. S. Arthanari when he was at Stanford from May to July, 1985. Emma has the source code but does not want to distribute it further without his knowledge.

The Font

There are approximately 160 characters in several styles written in a pre-release version of the current METAFONT. Emma has only tried producing characters for one style but had little difficulty in doing so (a few commands had changed). They are rough but look fairly good.

Contact

T. S. Arthanari's last known address was:
Quality Informatics Labs, Ltd.,
312, P. M. G. Complex,
57, South Usman Road,
Madras, 600 017, India.

I wrote to Mr. Arthanari in December 1987 to ascertain his intentions concerning his work, and especially to learn whether he is willing and able to allow the source code of his Tamil font to be distributed as public domain software. There has been, as yet, no reply. A colleague is visiting Madras in a few weeks, and will try to make contact with him.

4.2 Ramanujan

The Font

According to information received by e-mail from Pierre MacKay (Fri Jan 22, 1988), Ramanujan, a graduate student who worked at Washington two years ago, designed a Tamil font in METAFONT84 (I think). According to Pierre, the problem with this, as with Ghosh's Devanāgarī, is that it was arbitrarily developed in a framework that bears no relation to the monotype-based character grid used for Computer Modern, i.e., the characters do not sit in a box or on a baseline that relates in an appropriate manner to CM; this is unfortunate, since it makes it almost unusable in an environment with CM. Moreover, it does not make much use of the macro capabilities of METAFONT.

4.3 Ridgeway and Schiffman

After a thorough evaluation of the Ramanujan characters, the Humanities and Arts Computing Center at the University of Washington decided that it was better to begin again. Dr. Thomas Ridgeway, director of the Center, in consultation with Prof. Harold Schiffman of the Department of Asian Languages, has almost completed a fully vowelled METAFONT for Tamil. It will be tested during the summer, and should be ready for release in early fall 1988, together with some sort of macro package to make it usable from a Latin-letter keyboard.

Characters are arranged in the font in rough alphabetical order, starting from position zero. We have not found any other coding system which seems definite enough to use as a model. The Tamil phonetic subset of ISCII does not provide nearly enough character positions. It is expected that with this as with many other non-Latin fonts some sort of input preprocessor will be used to mediate between the code used for text-editing and the TEX font coding.

Contact

Dr. Thomas Ridgeway, Director
Humanities and Arts Computing Center
Mail Stop DW-10
University of Washington
Seattle, WA 98195, USA
Phone: (206) 543-6259
Net: `mackay@june.cs.washington.edu`

4.4 Other Developments

T. K. Rengarajan, a software engineer (Database Systems, Digital Equipment Corporation), e-mailed me on Fri 5 Feb, 1988, and mentioned that he may do a METAFONT Tamil. He can be contacted at:
Net: `ranga%debit.DEC@decwrl.dec.com`

5 Telugu

5.1 Mukkavilli

The Font

Recent mail from Lakshmi Mukkavilli indicates her intention, together with her husband Lakshmankumar Mukkavilli, to implement a Telugu script font in METAFONT. Their work on this font will intensify during the summer of 1988. Incidentally, they are looking for a good Telugu calligrapher and would welcome suggestions. In their opinion, existing Telugu typefaces are not good, tending to be very heavy along the baseline, which is hard on eyes. So they are not keen on imitating existing typefaces, and will probably create their own.

A note from Lakshmi in May 1988 said that she and her husband have now started working full time on the Telugu fonts using METAFONT, and they expect to start coding METAFONT programs in the second week of June. They are at present deciding on parameters, and the characters for which METAFONT programs are to be written. They are also concerned about how to incorporate context analysis logic in the form of TEX macros. Entering text in Roman transliteration and preprocessing the text for input to TEX does not seem very attractive to them, although, in the absence of widely available customizable terminals that could cope with Telugu, they may be forced down this path.

Another point of concern is their desire to use the ISCII standard (a version of ASCII for Indian languages). This is the only really widely published standard for the arrangement of the characters of Indian languages in a font grid, and I suspect that all creators of Indic fonts should be using it as far as possible.

Contact

Lakshmankumar Mukkavilli or
Lakshmi Mukkavilli,
226 Computer Science,
Iowa State University,
Ames IA 50011, USA.
Phone: (515) 296–7808
CSnet: `lakshmi@atanasoff.cs.iastate.edu`
Uucp: `lakshmi!atanasoff`

5.2 Other Developments

E-Mail from K. Sankara Rao in March 1988 indicates his similar intention to implement a Telugu font in METAFONT. He can be contacted at:
Department of Electrical Engineering
North Dakota State University
Fargo, ND 58105, USA.
Bitnet: `nu043109@ndsuvm1`

6 Perso-Arabic

6.1 MacKay

Pierre A. MacKay (TUG Site Coordinator for Unix-flavored TEX) and the Washington team have been working on an Arabic implementation of TEX for some years. Their plans are ambitious, and include building a customized version of TEX, called TEX-XET, which has a built-in capability for handling bidirectional text. Details of this change to TEX were published by Don Knuth and Pierre MacKay in *TUGboat* **8**, issue 1. This is an active project, but MacKay says wistfully that Arabic remains a long-term dream.

Terms of Availability

TEX–XET can be compiled with the C compiler using WEB-to-C and the change file `Cxet.ch`, both of which are part of the Unix TEX distribution. Several sites have actually made use of the reflection primitives, and Larry Denenberg of BBN has rewritten the **dvi2ps** device driver to do the correct things with right-to-left text. The Denenberg **dvi2ps** is vastly superior to the old **dvi2ps** in many other ways, and it will become a part of the distribution very shortly. (The driver is available as an FTP file from the `pub` directory on `june.cs.washington.edu`. Look for `ld_dvi2ps.tar.Z`.) WEB-to-C is also available as the file `web2c.tar.Z`, along with `Cxet.ch.Z`.

Contact

Pierre MacKay,
Department of Computer Science, FR-35,
University of Washington,
Seattle, WA 98195, USA.
Phone: (206) 545-2386/543-6259.
Net: `MacKay@June.cs.Washington.edu`

6.2 Goldberg

On Monday, 18 Jan 1988, and again on 15 Feb 1988, Jacques J. Goldberg wrote to TEXhax (1988, issues 7 and 15), giving details of a package giving the capability of printing Hebrew. He said that an article is currently being written about the package, for submission to *TUGboat* (see under **Hebrew**). At the same time he included a brief note referring to a nearly completed Arabic font.

The Font

Goldberg said,

> An Arabic font is three characters away from completion, but the MetaFounders are near midyear exams and unpaid, so the Arabic font *might* show up around mid March [1988]. (To be precise, their font is Parsi, and some limited work is needed to extend it to full Arabic). ... This needs *no* change either in TEX or in DVI drivers: a simple preprocessing of the TEX input file and a small additional macro package do it.

Terms of Availability

It is likely that the Arabic fonts and macros may be distributed on similar terms to the Hebrew; see the **Hebrew** section.

Contact

For Prof. Goldberg's address see under **Hebrew**.

6.3 Other Developments

See also the bitmap Arabic (?) fonts distributed by the **Austin Code Works**.

7 Hebrew

7.1 Goldberg

On Monday, 18 Jan 1988, and again on 15 Feb 1988, Jacques J. Goldberg wrote to TEXhax (1988, issues 7 and 15), giving details of a package giving the capability of printing Hebrew. He said that an article is currently being written about the package, for submission to *TUGboat.*

The Font

Goldberg says that the package comprises:

- a set of fonts at 8, 9, 10, 12, 17 points in regular type, 10 points slanted and bold, and any magnification on request (1000 off the shelf).
- a 100% portable preprocessor written in C (MSDOS users who do not have a compiler can get the `.COM` file).
- a small set of TEX macros.
- a sample file.

Usage

Hebrew words in Roman transliteration are inserted either by typing first-typed-last-read with the font invoked, which is a pain but 'displays' in natural reading order, or by typing first typed first read as argument of the `\reflect` macro given by D. Knuth and P. MacKay, *TUGboat* **8** (1987), p. 14. Long Hebrew sequences are typed, in first-typed-first-read order, within delimiters. The preprocessor copies non Hebrew sequences to an auxiliary file. Hebrew sequences are parsed into words, and written to the auxiliary file one word at a time after each word has been reflected. TEX is then invoked on the file containing the macro package, which itself `\inputs` the auxiliary files, feeding TEX with either normal English input or `\lines{ }` adjusted by the macro to the optimal number of Hebrew words.

Quality

Goldberg is — I suspect unnecessarily — diffident about the quality of the fonts. He calls them 'ugly fonts not good for anything else than Office documents (drafts, reports, ...)'.

Future development

Goldberg is looking for a convenient table representing the 22 Hebrew letters by Roman letters. Then the preprocessor could translate to standard ASCII the character codes used in Israel with their special Hebrew terminals, so that anybody with an English-only terminal could write in Hebrew.

Terms of Availability

Goldberg says,

> I do not expect any fee from individuals, but I would be happy if *institutions* that may use this package would later voluntarily contribute $25 to $50 [payable to the Treasurer of the University] to help my Department ... pay students employed on font development.

He later added,

> All that project is stored and freely available on a Bitnet server. To get the whole package, send an interactive message `GET IVRITEX PACKAGE to LISTSERV@TAUNIVM`. Arpanet or other nets not interactively connected to Bitnet, just send a `MAIL` file to that address, with the request `GET IVRITEX PACKAGE` in the first line (*not* Subject) of the message. Other useful commands to that server are:
>
> `GET IVRITEX FILELIST` to get a directory of the project, and
>
> `INFO FILES` to get instructions how to sign up for automatic updates.

Contact

Prof. Jacques J. Goldberg,
Department of Physics,
Technion-City,
32000 Haifa, Israel.
Bitnet: `phr00jg@technion`
If you are not on Bitnet, try:
`phr00jg%technion.bitnet` at
`forsythe.stanford.edu`

7.2 Other Developments

See also the experimental Hebrew font described under the heading **Georgia Tobin**, and the bitmaps distributed by the **Austin Code Works**.

8 Greek

8.1 Levy

The Font

Regular, bold, and typewriter versions of the Greek alphabet have been coded in METAFONT84 by Silvio Levy of Princeton, starting from the Greek character set created by Don Knuth as part of the CM family, but with all accents, breathings, correct spacing, ligatures, and macros to implement a convenient Roman transliteration for input. The font is suitable for both classical and modern Greek.

Full details and illustrations of the use of the fonts have been given by Silvio Levy in his two publications:

'Typesetting Greek', in *TEX Users Group Eighth Annual Meeting: Conference Proceedings*, edited by Dean Guenther (Providence: TUG, 1988), 27–33.

'Using Greek Fonts with TEX', *TUGboat* **9** (1988), 20–24.

Terms of Availability

Silvio Levy has released his Greek fonts freely, without charge (and without any warranty). He maintains an electronic mailing list of interested parties, and the METAFONT source is available to Arpanet users by anonymous FTP from `princeton.edu`.

Contact

Silvio Levy,
Math Department, Fine Hall,
Princeton University,
Washington Road,
Princeton, NJ 08544, USA.
Phone: (609) 452-5790
Net: `levy@princeton.edu`

8.2 Kelly

Issue 14 of UKTEX 1988 (Fri, May 20), the UK's answer to TEXhax, put out from Aston by Peter Abbott (`abbottp@aston.ac.uk.bitnet`), carried a notice by Christopher P. Andrasic (of Cranfield) reporting some Greek fonts.

The Fonts

Brian Hamilton Kelly (also at Cranfield) created these Greek fonts using METAFONT84. The METAFONT sources of the non-Math Greek fonts are contained in the files `CMGI10.MF`, `CMGTT10.MF`, `CMG10.MF`, `CMG810.MF`, and `GRKTXT.MF`. As far as I know, there are no macros or preprocessor offered for implementing the font at the keyboard level.

Terms of Availability

These fonts are being offered for general, free release. They are available for FTP within the UK, from the directory `public.mffiles` of the UKTEX archive at Aston.

Contact

Christopher P. Andrasic
Net: `rm001a@uk.ac.cranfield.cdvc`

9 Cyrillic

9.1 MacKay

Pierre MacKay reports that the Washington team is working on Old Russian (more or less Old Church Slavonic, but specifically designed for the Slovo).

Contact

For Prof. MacKay's address, see under **Arabic**.

9.2 MF Slavic Family

This family of Cyrillic fonts is described under **Georgia Tobin**.

9.3 AMS

The American Mathematical Society has developed a post revolution Cyrillic font, in old METAFONT79, and a set of macros to implement it comfortably. Details of the font, with examples of its use, and grids of the character set were published in *TUGboat* **6** (1985), 124 ff.

Terms of Availability

Same terms and contact addresses as described under **AMSFonts**.

9.4 Other Developments

See also the Cyrillic bitmap fonts distributed by the **Austin Code Works**.

10 Turkish

10.1 Washington

The Font

Pierre MacKay informs me (Jan 22 1988) that work on properly accented Roman-letter Turkish fonts in METAFONT has been undertaken at the University of Washington by himself and Walter Andrews. The accented characters are developed from Computer Modern descriptions, so as to maintain the maximum possible compatibility with the Computer Modern faces. Andrews and MacKay have published a description of their work as:

'The Ottoman Texts Project' in *TEX Users Group Eighth Annual Meeting: Conference Proceedings*, edited by Dean Guenther (Providence: TUG, 1988), 35–52.

Pierre also wrote on 'Turkish Hyphenations for TEX' in *TUGboat* **9** (1988), 12–14. See also the note in *TUGboat* **8** (1987), 260.

Contact

For Prof. MacKay's address, see under **Arabic**.

11 Japanese

In order to typeset Japanese text it is insufficient merely to have a Japanese font. There are several problems, including the very large size of the Japanese character set, which mean that a modified implementation of the TEX system as a whole is necessary.

Some of the issues concerned in using TEX for typesetting Japanese were surveyed by Nobuo Saito and Kazuhiro Kitagawa of Keio University, Yokohama:

'What Should We Do for Japanese TEX', in *TEX Users Group Eighth Annual Meeting: Conference Proceedings*, edited by Dean Guenther (Providence: TUG, 1988), 53–56.

Saito and Kitagawa have taken Pat Monardo's Common TEX as the starting point for a series of modifications to TEX itself that have some features in common with the ASCII Corporation's work, described below.

I recently received a comprehensive message from Edgar M. Cooke about the current state of TEX in Japan. Most what follows is reproduced verbatim from this message.

At present, two publicly available versions of TEX are being distributed that support Japanese, and one further version is supposed to become available shortly. These are not entirely mutually compatible, and each has its strong and weak points.

11.1 ASCII Corporation

The ASCII Corporation is a microcomputer oriented publishing and software house that has taken a strong interest in TEX for their own publishing work, and whose UNIX support section has produced and distributes the Japanized version of TEX.

ASCII's version was implemented without concern for 100% internal compatibility with Knuth's TEX, and cannot pass the `TRIP` test (N.B.: it is closer to passing as of version 1.0). However, the output of an identical English TEX input file is, to the best of my knowledge, identical with that of standard TeX. They have added a few primitives (concerned with spacing between the ideographs and letters, etc.) useful in handling Japanese or Japanese with Romanized languages, and have created a variant of TFM which they call JFM (but which still has the TFM extension, although it is internally identifiable by a coded ID byte). The purpose of this is to allow one font to hold the more than 6000 characters of a typical Japanese font. It also differs in other ways, notably that the ligature table (unnecessary in Japanese) has been replaced by a 'glue table', which handles much of the information necessary concerning spacing between the characters (which usually appear without distinction between inter-character and inter-word spaces, unlike in modern Western languages).

Terms of Availability

At present, the ASCII Corporation is freely distributing this implementation in the form of sources including change files for INITeX, VIRTeX, and BiBTeX, and Japanized macro files for plain TeX, LaTeX, and SliTeX (with the LaTeX style files), and they include the source for a (just adequate) printer driver for the Canon LBP-8 that uses its internal Japanese character set, and JFM files for point sizes 5 through 10 of a standard ('Mincho' = Ming Dynasty style) typeface and of an emboldened ('Gothic') typeface which can probably be used with a number of different pixel font sets, e.g., the internal Canon LBP-8 set, (but which in ver. 1.0 is based on the Dai Nippon Printing Co. [DNP] fonts). A printer driver and X-windows previewer handling the DNP fonts is now available.

Contact

ASCII Corporation,
Sumitomo Minamiaoyama Building,
5-1-5 Minamiaoyama Minato-ku,
Tokyo 107, Japan.

11.2 JTeX

The following articles on JTeX and its fonts, by Yasuki Saito, have appeared:

'Japanese TeX', *TUGboat* **8** (1987), 103–116.

'TeX: JTeX', in *TeX Users Group Eighth Annual Meeting: Conference Proceedings*, edited by Dean Guenther (Providence: TUG, 1988), 57–68.

JTeX was developed by Yasuki Saito of Nippon Telephone & Telegraph (NTT). Saito's policy has been to attempt to avoid radical changes to standard TeX as much as possible, but this increased compatibility has led to problems: since a font can have only 256 characters, the number of fonts one needs even to support a single real Japanese font set (33) tends to be quite large, even if one only declares the fonts corresponding to the various sections of the original font that include characters that have actually been input into the document.

A standard (but ugly) set of Japanese characters that is in the public domain (known as the JIS fonts, for the Japanese Industrial Standards Institute, which is responsible for the (abjectly arbitrary) standard coding of Japanese characters and for making the font set available) is included with JTeX. This originally consisted of only 1 size, namely characters described in a 24-dot square matrix, but Saito mechanically generated 36-, 48-, and 72-dot fonts to imitate other point sizes or `\magsteps`. This set is not very high quality by any standard, but it is the only public domain font known to me [Cooke].

Yasuki Saito has also collaborated with Dai Nippon Printing Co. to make their industry standard fonts available — but for a price: 95,000 Japanese yen.[7]

Saito's 1988 article, cited above, includes a section describing the JIS and DNP Japanese fonts, with illustrations.

JTeX has 240, 300, 400, and 480 dpi fonts available currently. Please note that 6000+ characters makes for a largish distribution tape — the rudimentary set of JIS fonts in an adapted GF format with TFMs takes up about 10 M of disk space, while the PXL, GF, and TFM files for the DNP Mincho and Gothic fonts very nearly fill an entire 2400 ft magtape at 6250 bpi! I had a chance to measure these more carefully:

DNP fonts		JIS fonts	
GF Mincho	49,312,713	PXL	11,321,384
GF Gothic	43,401,652	GF	7,729,124
TFMs	559,152	TFMs	137,417
Total bytes	93,273,517		19,187,925

In other words, non-trivial.

Terms of Availability

JTeX is in the public domain, and comes free with the JIS fonts. In Japan, the Japan Society for Software Science and Technology distributes it.

[7] ASCII and SONY are also negotiating with Dai Nippon to allow similar font sets to become available for different resolutions of printers, and, if we are lucky, a vector stroke typeface of high quality that is applicable to various sizes and resolutions may become available next year.

JTEX is is available in the US by anonymous FTP from turing.stanford.edu. The Tops-20 version is installed on Turing in directory PS:<JTEX>.

The UNIX version is being distributed for the time being by Yasuki Saito, with source files for pretty much what ASCII has, but with an Imagen printer driver that handles external Japanese font data and an X-windows previewer for JTEX, as well as a version of DVI2PS supporting Japanese, but no support for BiBTEX or SliTEX is offered.

But in a note to TEXhax 1987, issue 106 (Fri 25 Dec), Hideki Isozaki announced that he has prepared LATEX and SliTEX files to work with JTEX.

Contacts

Yasuki Saito,
NTT Electrical Communications Laboratories,
NTT Corp., 3-9-11
Midori-cho Musashina-shi,
Tokyo 180, Japan.
Phone: +81 (422) 59-2537
Net: yaski%ntt-20@sumex-aim.stanford.edu

Hideki Isozaki,
NTT Software Laboratories,
JUNet: isozaki@ntt-20.ntt.junet
CSNet: isozaki@ntt-20.ntt.jp
Arpa: isozaki%ntt-20@sumex-aim.stanford.edu

11.3 Other Developments

There is a version of Japanese TEX that is likely to become available from Canon which resembles JTEX, but it may take a *long* time for the management at Canon to make anything available to the public domain, so it may not be available until some time next year.

Similarly, IBM Japan has a Japanese version based on the Canon algorithm (which was also the stimulus for the Saito's JTEX).

The Bitstream Kanji fonts could almost certainly be integrated into the above Japanese TEX implementations (see **Bitstream**).

11.4 Conclusion

There has been a meeting of the leaders and interested parties of J-TUG about whether it is possible to merge these three versions, and a number of suggestions and guidelines have been set forth. But lacking a central authority figure along the lines of a Donald Knuth has limited this from going beyond recommendations.

All three parties are working on improving the inherent quality and mutual compatibility of their versions, and I expect that they will be working on this throughout the next year (all this being more or less volunteer work). Except for the fact that just about none of the other utilities (DVITYPE, etc.) seem to work with the ASCII version (except TFtoPL and PLtoTF to preen JFMs), I suspect that it is a leading contender, because

- they are promoting it through prompt and widespread distribution of the sources advertised in their own and others' publications, and
- because they have obviously lavished a good deal of care in attending to details of Japanese printing practice that make it at least as good as either of the others — in addition, of course, to removing the burden of having to deal with a plethora of fonts each comprising a tiny fragment of a whole Japanese font set.

Contact for all Japanese TEX

Edgar M. Cooke is prepared to act as a clearing house for TEX going into and coming out of Japan. He can be contacted at:
Software Research Association Inc.,
1-1-1 Hirakawa-cho,
Chiyoda-ku,
Tokyo 102, Japan.
Net: cooke%srava.sra.junet@uunet.uu.NET[8]

Edgar Cooke is in close touch with Pierre MacKay, and has sent Pierre the two versions of Japanese TEX (ASCII Corp. and JTEX) mentioned above.

12 Chinese

Work done on a Chinese METAfont by Gu Guoan and John D. Hobby is available by anonymous FTP from june.cs.washington.edu, in the directory /pub, as the (large) file CHINESE.TAR.Z. This was written up in *TUGboat* **5** (1984), pp.119–136. (This is a reprint of the Stanford Computer Science Report 974 by Gu Guoan and J. D. Hobby: *A Chinese Meta-font* (Stanford, 1983).)

13 International Phonetic Alphabet

13.1 Washington State University

Dean Guenther informs me (June 22, 1988) that Washington State University has an IPA font available. It contains 128 popular IPA characters and diacritics as specified in the *Phonetic Symbol Guide*

[8] N.B. It is possible to reply to any mail he will send you by 'R' or 'r' from the Unix mail(1) program, but then it will come via CSnet, which (although days faster) is prohibited to non-member, non-academic institutions such as his.

by Geoffrey K. Pullum and William A. Ladusaw (Chicago, London, 1986). Janene Winter did the METAFONT work on this font. The character positions were coordinated with help from Helmut Feldweg at the Max-Planck-Institut für Psycholinguistik in the Netherlands, Christina Thiele at Carleton University and some ideas from Brian MacWhinney at Carnegie Mellon and Karen Mullen at the University of Kentucky at Louisville.

The font also comes with a set of macros to access the characters easily. For example, `\schwa` prints what you would expect.

Terms of Availability

The Washington State University IPA is available for $100. The package includes `GF`, `PXL` or `PK` fonts at 9, 10, 11 and 12 point (together) in the Roman bold and slanted faces (together). The typeface is designed to match the CM Roman face. The METAFONT source is not included.

Contact

Send a note to Dean Guenther at `guenther@wsuvm1` on Bitnet, or write to:
TEXT1 Distribution,
Computing Service Center,
Washington State University,
Pullman, WA 99164-1220, USA.

13.2 Other Developments

Georgia Tobin (q. v.) has an IPA font, created in old METAFONT79.

A bitmap IPA font, `ph10`, was created by Jean Pierre Paillet for use with TEX for typesetting the *Canadian Journal of Linguistics*. This font is described, with a printout of the character grid, by Christina Thiele in

> 'TEX, Linguistics, and Journal Production' in *TEX Users Group Eighth Annual Meeting: Conference Proceedings*, edited by Dean Guenther (Providence: TUG, 1988), 5–26.

`ph10` is now superseded by the Washington State University font.

According to a note from G. Toal in UKTEX 1988, issue 2, Tibor Tscheke's company, Stürtz AG, also has an IPA font for sale. Toal does not state whether this font was created with METAFONT, but the implication is that it is usable with TEX. Contact:
Tibor Tscheke,
Head, Computer Science Department,
Universitätsdruckerei,
H. Stürtz AG,
Beethovenstraße 5,
D–8700 Wurzburg,
West Germany.

Kris Holmes and Chuck Bigelow also report that they have a bitmap IPA font. See **Lucida**.

13.3 Ridgeway

A phonetic alphabet has been developed by Thomas Ridgeway for a large subrange of American Indian languages. The first active projects using this are in Salish and Navajo. This font is presently being tested and will be available from the Humanities and Arts Computing Center at the University of Washington in early fall 1988.

Contact

See under **Tamil** above.

14 Elvish

Elvish, or more properly, Tengwar, is the script used by the elves of Middle Earth. It was described by J. R. R. Tolkien.

14.1 Urban

On 4 Dec 1986 (*sic*) Mike Urban released the METAFONT code for the Tengwar script through the Usenet newsgroup `comp.text`.

The Font

Mike said the following:

> [Here are the] METAFONT sources for a digitized version of the Tengwar (Elvish script) created by J. R. R. Tolkien. They have only been tested on a 300dpi laser printer. No guarantee of the quality of either the code or the output is offered. I'm not particularly satisfied with the quality of the code (my first non-trivial attempt to use METAFONT), but the results look OK to me.

Contact

Michael Urban,
TRW Inc., R2/2009
One Space Park,
Redondo Beach, CA 90278, USA.
Phone: (213) 812-0632
Net: `urban@spp2.UUCP`

14.2 Other Developments

See also the Elvish bitmap fonts available from the **Austin Code Works**.

15 Georgia Tobin

Georgia Tobin is well known to readers of *TUGboat* as the editor of, and chief contributor to, the 'Font Forum' section of the journal. She has been working with METAFONT since 1982, and between 1982 and 1987 she created several complete families of fonts for use with TEX. (Georgia's husband Rick works with her on the fonts.) Because much of her earlier work was done using METAFONT79, which is now superseded, only the bitmaps of these early fonts are available but not the METAFONT source code.

The bitmap fonts are mostly at 300dpi, and are optimized for write-black imaging machines (i.e., Apple, Canon, HP, and certain other laser printers). The fonts are available at a wide range of sizes ranging from 5 to 72 points (some of the less common fonts are available in a narrower range of sizes, say from 5 to 36 point). One particular subset of these fonts, marketed by Personal TEX Inc. as a package called MF Medley, consists of the Chel fonts at 5, 7, 10 and 12 point sizes, with Copperplate and Schoolbook at 36 and 48 point, and Black Letter at 36 point. The MF Medley is available at 180, 240 and 300dpi, with some fonts available at 118dpi too, and costs $100.

An important point to notice is that the Roman, Chel and Schoolbook families described below include math symbol and extensible fonts, like Knuth's CM, so that these fonts can be used for the full range of mathematical and technical typesetting as defined in the `PLAIN` format and LATEX.

Georgia's newer work on Schoolbook, Hebrew, ALA and Special Effects typefaces, described below, is all done in METAFONT84, the current and stabilized version of METAFONT, and is therefore much more flexible. Fonts of these faces can be generated at any reasonable resolution, and for any marking engine with a defined `mode`. One hopes that Georgia will find some way of making her METAFONT source code available to bona fide users of her newer typefaces, without of course jeopardizing her livelihood.

15.1 MF Chel Family

The Chel ('Computer Helvetica') family of sans-serif fonts was initially created by Thom Hickey in a Tandem TAL translation of METAFONT79 (later recoded into Apollo Pascal using the MAP preprocessor). He began work on the font in the winter of 1980, and continued to work on the font until 1982.[9] Chel was later completed and extensively reworked by Georgia Tobin. In its finished form, Chel has been described as 'lighter and more compact' than the Computer Modern sans-serif (`CMSS`) which was designed by Richard Southall and is included in all CM distributions. Chel comprises fifteen fonts including Chel Book, Slant, Medium Bold, Slanted Medium Bold, Bold, Slanted Bold, Extra Bold, Slanted Extra Bold, Math Symbols, Bold Math Symbols, Math Italic, Bold Math Italic, Math Extensible, Elite, Bold Elite, Pica and Bold Pica.

15.2 MF Roman Family

This family comprises more than nineteen Times Roman style seriffed fonts, including Roman Text, Slanted Text, Italic, Unslanted Italic, Medium Bold, Medium Bold Italic, Bold, Bold Italic, Extra Bold, Extra Bold Italic, Titling (Small Caps), Slanted Titling, Math Symbol, Bold Math Symbols, Math Italic, Bold Math Italic, Math Extensible, Elite and Pica. The last two fonts are 12 cpi and 10 cpi typewriter style fonts respectively, and include slanted and emboldened versions.

15.3 MF Slavic Family

The Slavic Family of fonts includes all the fonts necessary for sophisticated typesetting in Russian. The family includes Chel-compatible and Roman-compatible versions of Cyrillic in Book, Slant, Bold and Bold Slant versions, and also an Italic version of the Roman-compatible face. There are further 'additional' fonts corresponding to each of these categories which contain extra accents and characters used in typesetting other Slavic languages.

15.4 MF Decorative Family

Also offered is a decorative package of fonts which includes six typefaces, including Black Letter, a Copperplate Script, Hodge Podge (including assorted dingbats, pharmacy and planetary symbols, a turtle and a frog), an Outline Helvetica (upper case), and a Slanted Outline Helvetica (also upper case), and an Uncial Majuscule which emulates a medieval manuscript script.

15.5 Century Schoolbook

Georgia is close to finalizing a first release version of a Century Schoolbook typeface. This is the first fruits of her work with the new METAFONT84. Her goal has been to create a complete Century Schoolbook style typeface that is clean and legible from

[9] Reference to Hickey's work on Chel, with an illustration of the MF code for the letters 'B' and 'b', is made in Knuth's 'Lessons Learned from Metafont' (1985), 37–38.

very low resolutions (about 72dpi is the lowest so far) to very high, and in point sizes from 5 to 96 or so.

15.6 MF ALA

Another project has been the creation of METAFONT fonts which include the special character set defined as a standard by the American Library Association (ALA) and used by the Library of Congress and other bodies influential in the library automation world, such as OCLC. This set of characters and accents was designed to make possible the representation, if necessary in a standard Roman transliteration, of virtually all the world's languages. The characters include items like upper and lower case thorn, Polish dark *el* (with a cross bar), eth, and several other unusual signs and accents. There is even a *candrabindu* for Sanskritists! Georgia did this work for the Library of Congress. The font is available in Text, Bold, Italic, and Bold Italic.

The ALA fonts contain 256 character positions, and therefore require a robust DVI driver program such as the members of Nelson Beebe's DVI driver family or the latest release of the Arbortext drivers.

15.7 Hebrew

Georgia is developing a Hebrew typeface, which is still at an experimental stage. At the present time it consists of 27 characters, with more calligraphic molding of the strokes than is shown in Goldberg's font. However, Georgia has not developed any macros or preprocessors for inputting Hebrew text in quantities, as Goldberg has. It would be advantageous if Georgia and Goldberg were to standardize on a common font layout, so that any macro/input system would be able to access either of their fonts.

15.8 Special Effects

Georgia published 'The ABC's of Special Effects' in *TUGboat* **9** (1988), 15–18, in which are demonstrated several fascinating typographical effects that are relatively simple to produce with clever use of METAFONT84 macros. The article includes the METAFONT code illustrating how the effects were produced.

Terms of Availability

The Chel, Roman, Slavic, Decorative and Century Schoolbook font families are available on a commercial basis from the following sources:
ASCII Corporation,
Sumitomo Minami Aoyama Bldg.,
5–11–5 Minami Aoyama,
Minato-ku,
Tokyo 107, Japan.

Docusoft Publishing Technologies,
Suite 300,
1120 Hamilton Street,
Vancouver, B. C. VV6B 2S2,
Canada.

Interbase,
Dantes Plads 1,
DK–1556 Copenhagen V,
Denmark.

Personal TEX, Inc.,
12 Madrona Avenue,
Mill Valley, CA 94941,
USA.

TEXpert Systems Ltd.,
5 Northernhay Square,
Exeter EX4 3ES,
Devon, UK.

Georgia has some fine catalogues illustrating the Roman, Chel and Decorative families: send a cheque for $6 ($15 outside USA or Canada) to:
Georgia Tobin,
1888 Barnard Drive,
Powell, Ohio 43065.
Phone: (614) 764–9863.

16 Blackboard Bold

16.1 Robert Messer

Robert Messer published an article 'Blackboard Bold' in *TUGboat* **9** (1988), 19–20, in which he generalized a method used by Knuth in *The TEXbook* to produce such characters. This is a series of Plain TEX macros which jiggle the characters of CM around, using small kerns and the capital I and small rules, to produce a 'poor person's blackboard bold'.

16.2 Other Developments

See also **Custom fonts & Pandora**, and the **AMSFonts package**.

17 APL

The APL programming language requires many unusual symbol characters, which often baffle normal typesetting and word processing systems.

17.1 Hohti and Kanerva

Aarno Hohti and Okko Kanerva of the University of Helsinki have developed an APL font for use with TEX. They have 'raided' the CM character set to this end, so the characters should be similar in weight and style with CM.

The font is described in the article 'Generating an APL Font' in *TUGboat* **8** (1987), 275–278.

Terms of Availability

The authors can be contacted at:
University of Helsinki,
Department of Mathematics,
Hallituskatu 15,
SF-00100 Helsinki,
Finland.

I assume that the fonts are being freely distributed, since the METAFONT code for them is available for anonymous FTP as file `CMAPL10` from `score.stanford.edu`, in directory `<TEX.TUGBOAT>`.

18 AMSFonts Package

The American Mathematical Society (AMS) has developed several fonts of symbols and alphabets intended for use in mathematical notation.

18.1 The Fonts

Three alphabets, collectively known as Euler, were designed by Hermann Zapf and implemented in METAFONT at Stanford as part of the TeX project. They come in both medium and bold weights, and include Fraktur, script and an upright cursive alphabet, which was intended to minimize problems with the placement of accents and indices. The Euler fonts are considered proprietary, and sources are available only under lease.

Two fonts of symbols, including a Blackboard Bold alphabet, are also available. Details of these fonts, including character grids, were published in *TUGboat* **6** (1985), 124 ff. These fonts are still rendered in METAFONT79, which is totally incompatible with the current METAFONT, so distribution of the sources is pointless; arrangements are being made for their re-implementation in new METAFONT, but the schedule is uncertain.

18.2 Terms of Availability

Together with Cyrillic (see under **Cyrillic**) fonts, this collection is called AMSFonts. A set of `TFM` and `300GF` files (`magstep0` only) is available:

- by anonymous FTP from directory `<TEX.AMSFONTS>` at `score.stanford.edu`;
- as part of the standard distribution from Maria Code;
- as part of other major TeX distributions.

The AMSFonts are available from the AMS in a full range of magnifications and in additional resolutions on IBM PC-compatible and Macintosh diskettes and on mag tape in VAX/VMS format. For information, contact:
AMS TeX Library,
American Mathematical Society,
P. O. Box 6248,
Providence, RI 02940, USA.
Phone: (401) 272-9500 or (800) 556-7774
Internet: `sse@math.AMS.com`

Contact for technical inquiries:
Barbara Beeton,
(same address)
Phone: (401) 272-9500
Internet: `bnb@seed.AMS.com` or
`bnb@xx.lcs.MIT.edu`

19 Custom Fonts & Pandora

19.1 Custom Fonts

If you are desperate for a TeX font that does not yet exist, why not commission a METAFONT programmer to create it?

Neenie N. Billawala advertises her services as a METAFONT consultant in *TUGboat*. She is responsible for creating the fine calligraphic capitals that are part of the Computer Modern typeface family (in the `CMSY` fonts).

19.2 Pandora

Neenie, a designer, has also created a new typeface called Pandora, which is part of a larger research project concerning the possibility of breaking the elements of typeface design down into general reusable components such as serifs (and terminal endings), bowls, circular shapes, arms and so on. Pandora is the result of setting the parameters for these components to one particular set of values, but many others could be chosen. In this sense, Pandora explores further the 'parametrization' which is at the heart of Knuth's endeavor with METAFONT. Neenie has nearly finished writing a Stanford Computer Science Report about this, called *Meta-Marks: Preliminary Studies for a Pandora's Box of shapes* (to appear). Knuth describes this study as 'lavishly illustrated studies in parameter variation, leading to the design of a new typeface called Pandora'.[10]

The Pandora typeface is intended to be a 'bread and butter' text face and has been generated in seriffed, sans-serif and fixed width versions. The character set of Pandora coincides with `CMR` and `CMTT`,

[10] *Computer Modern Typefaces* (Reading etc., 1986), xiii.

etc., and thus it does not include the math symbol characters and extensibles.

Terms of Availability

Neenie hopes, in the longer run, to donate Pandora to the TeX community, perhaps submitting it for inclusion in the standard distributions.

19.3 Blackboard Bold & Outline

Neenie is also working on a Blackboard Bold, to be compatible with Times Roman, for the AMS, and an outline font.

Contact

Neenie Billawala,
841 Stendhal Lane,
Cupertino, CA 95104, USA.
Phone: (408) 253-4833
Uucp: `(ihnp4, seismo, decwrl,`
`ucbvax, ... )!sun!metamarks!nb`

20 Bitstream Font Family

20.1 The Fonts

In my view, one of the most exciting developments in the area of Roman alphabet typeface availability for TeX has been the recent announcement by Personal TeX Inc., of the PTI Font Interface Package (FIP). This is an MS DOS program that converts the outline typefaces of the Bitstream typeface library into PXL (and then PK) fonts, with associated TFM files, for use with TeX in a manner analogous to the use of the use of the Computer Modern fonts.

Since the Bitstream typeface outline files for any given font contain more than the 128 characters usual in a TeX font, the extra Bitstream characters can be generated in a second, complement font. The Bitstream fonts generated by the FIP contain the same characters as CMR10. However, equivalents of the math italic, math symbol and extensible fonts of the CM family are not provided, so the Bitstream fonts are for use in typesetting predominantly textual matter. (One could, of course, mix the fonts, using Bitstream for the text and CM for the mathematics.)

The method of producing the font bitmaps is exactly the same, in principle, as using METAFONT, except that one has no access to the underlying character descriptions. The FIP reads the typeface outline data and generates bitmap fonts at any desired point size between 6 and 72 points and above, and at any resolution, from below 100dpi for IBM PC screens, to over 1000dpi for typesetters.

The Bitstream typeface library advertised by Personal TeX at present includes the following typefaces, each consisting of a regular, an italic, a bold and a bold italic face: Baskerville, Bitstream Charter, Bitstream Cooper Light, Century Schoolbook, Courier, Dutch (i.e., Times Roman), Futura Book, Futura Light, Futura Medium, Goudy Old Style, ITC Avant Garde Gothic, ITC Bookman Light, ITC Galliard, ITC Garamond Condensed, ITC Garamond, ITC Korinna, ITC Souvenir Light, Letter Gothic, News Gothic, Prestige, Serifa, Swiss (i.e., Helvetica), Swiss Condensed, Swiss Light, Zapf Calligraphic (i.e., Palatino), Zapf Humanist (i.e., Optima), and a selection of Headline faces including Bitstream Cloister Black, Broadway, Cooper Black, and University Roman.

New Fontware outline typefaces are regularly released by Bitstream: their catalogue currently has 40 faces.

Users of operating systems other than MS DOS can presumably use the Bitstream PXL or PK fonts (with TFMs) once generated on a PC/AT, just by uploading them with, say, Kermit.

20.2 Quality

The creation of the font bitmaps is done by the FIP using what Bitstream calls 'smart outlines'. Typographic rules are stored with the typeface outlines and are applied at the time of bitmap generation, using artificial intelligence algorithms (originally implemented on Symbolics 3600 Lisp workstations) to tailor significant features of the font to its point size, and the resolution and marking characteristics of the printing device. Some typographically significant features that are so treated are the stem weight, x-height, cap height, side bearings, and baseline alignment. So a 6 point Bitstream font is not a mere linear reduction of some larger design size, just as CMR6 is not simply a small CMR10. This is very significant for the high quality of the fonts at small or large sizes, and goes some of the way towards meeting the argument made by Knuth in *The TeXbook* (p.16), against scaling fonts much beyond their design size.

Optional software switches set when running the FIP permit the adjustment of accents and letter spacing. The newest release of the Fontware software also includes the choice of producing bitmaps appropriate to write-white as well as write-black marking engines. In the former case, the software will add a half-pixel layer all the way around a character to compensate for the erosion that occurs on white writers.

A recent discussion of fonts in *PC Magazine* (March 15, 1988, issue **7**(5), p. 238) noted that:

> Bitstream fonts are the same ones you get in already-generated form from H[ewlett] P[ackard], are widely (though not exclusively) used in the printing industry, and are used in our Tegra galley-generating machine.

A useful article comparing Bitstream fonts with Adobe PostScript fonts was published in the magazine *Publish!* (March 1988, issue **3**(3), pp.46 ff.). It included valuable illustrations of both manufacturers' fonts at several different resolutions.

I myself have only experimented with the Dutch and Swiss fonts at 10pt, and then only on a 640 × 400 pixel screen, and at 240dpi on a 9-pin matrix printer. On both these devices the Bitstream fonts appear *much* superior to the nearest CM equivalents, (CMR10 and CMSS10). The characters are somewhat broader, with relatively rounder bowls and shorter ascenders, and give a more even, regular appearance across the page. They make the CM fonts look very uneven by comparison (and I am a great fan of CM). I suspect that the CM fonts would compare more favorably at higher resolutions, since they were not designed for such poor output devices.

The Bitstream Font Interface Package offers TEX users (without access to PostScript printers) access for the first time to a proper typeface catalogue, and a highly professional one at that.

20.3 Other Developments

Although Bitstream has released 40 Fontware outline typefaces, it actually has a library of over 1000 digital typefaces waiting in the wings. This collection includes traditional designs, original designs, pi-fonts, and non-Latin fonts such as Arabic, Cyrillic, Greek, Gujarati, Hebrew, Laotian, and Tai Dan. However, all these exist in plain digital outline format only, which means that the outlines have not had the AI scaling optimization rules added to them, and are thus not 'smart outlines' such as can be used with the Fontware Installation packages. They are are thus not at present marketed widely.

Intelligent font scaling for Kanji fonts (Nippon Information Science Ltd. Iwata Gothic, with other faces to follow) was announced in March 1988, and is offered to OEMs.

20.4 Terms of Availability

The Fontware Installation Kit costs $195, and normally comes with the Swiss font family (one can make a special request to have the Dutch family instead).

20.5 Contacts

Personal TEX Inc.,
12 Madrona Avenue,
Mill Valley, CA 94941.
Phone: (415) 388-8853
Telex: 51060 10672 PCTEX
Fax: (415) 388-8865

Bitstream,
Athenaeum House,
215 First Street,
Cambridge, MA 02142.
Phone: (617) 497-6222
Telex: 467237
Fax: (617) 868-4632

21 Times Roman in METAFONT

21.1 Kemmish

The Font

The creator of the font, Ian Kemmish, writes:

> I have a Times font in METAFONT which I have been tinkering with over the past six or seven months. It is modelled on Monotype Times New Roman (visually — no calipers in sight!) The standard of rasterization is about comparable to a LaserWriter, but the typography is of necessity a lot more amateurish. The regular font is largely OK, the italic is a few weeks behind and probably needs some tweaking. There is a rather lumpy semibold which needs parameter tweaking. I suspect a genuine bold needs a new set of minuscule routines. I also have a typewriter font generated from it, and am working on a Nebiolo Eurostyle sans-serif font. (I needed something easy after the Times! I want to do Helvetica sometime soon.)

Terms of Availability

Ian Kemmish writes:

> I'd be happy to distribute what I have, though I suspect my employers would want to charge something for it. Ideally, I'd like to send out some GF files first to people who are interested in test-driving them and can make constructive comments about how to improve them. The METAFONT code is still in a state of flux. I'd hate to distribute it

> and have a lot of unco-ordinated changes being made to it all!

Contact

Ian Kemmish can be contacted at:
Whitechapel Workstations,
75 Whitechapel Road,
London E1 1DU, England.
Phone: (+44) 01 377 8680
Telex: (UK) 885300 WCW G
Fax: (+44) 01 247 4589
Uucp: `ian@wcw.co.uk`
OldUucp: ...`!mcvax!ukc!wcwvax!ian`

21.2 Other Developments

See also the MF Roman Family by **Georgia Tobin**, and the Dutch typeface by **Bitstream**.

22 Lucida

Lucida is the name of one of the typefaces designed by Kris Holmes and Chuck Bigelow. Its main design aim is that it be legible and beautiful at low as well as high resolutions, and it is probably the first original typeface family produced for digital printers and displays.[11] By low resolution, Holmes and Bigelow mean laser printers and computer screens. The font has been discussed in the following publications:

Jonathan A. Epstein, 'Best Font Forward', *Digital Review* (July 1986), 82–87.

C. Bigelow and K. Holmes, 'The Design of Lucida: an Integrated Family of Types for Electronic Literacy', in *Text Processing and Document Manipulation* edited by J. C. van Vliet (Cambridge, 1986), 1–17.

'Alumna Designs First LaserWriter Typeface', *Harvard Extension Newsletter* **8**(2) (Spring 1988).

The second of these articles is itself printed in the Lucida typeface. It goes into detail about the design concepts of Lucida, with illustrations.

22.1 The Fonts

The Lucida family includes the following eight fonts: Roman, Italic, Bold, and Bold Italic, in both seriffed and sans-serif styles. Lucida has been called a 'super family' because of the wide range of characters and fonts it provides, including compatibility with the full CM character set. Unusual features of Lucida fonts include the fact that the italic sans-serif is a true cursive style, rather than a slanted Roman, and that there are alternate sets of capitals, one heavier in weight, for English and French typographers, and one lighter, for Germanic texts which use extensive capitalization, and therefore need de-emphasized capitals.

The screen 'versions' of Lucida are at such low resolution (75–100dpi) that they cannot be regarded as straight reproductions of their higher resolution counterparts. They are therefore called Pellucida, to suggest that the designs are related to Lucida, but optimized for 'pel' based screen displays.

Adobe

Chuck Bigelow informed me in April 1988 that Adobe Systems is dealing with the release of several Lucida typefaces for use with TEX. Dan Mills, Manager of Typography at Adobe informed me later in the same month that:

> Knowing people would want these fonts for use with TEX (because of the Math versions, ...), we purposely extended our normal character set for these fonts (by about 16 characters) to cover the 'TEX text' set (Figure 1 in Appendix F of *The TEXbook*). What I mean is, these fonts have a union of our standard character set and the TEX text set.

The following faces have already been released by Adobe as downloadable PostScript fonts on both Mac and PC disks: Lucida Roman (seriffed), Lucida Italic, Lucida Bold, Lucida Bold Italic, Lucida Sans Roman (sans-serif), Lucida Sans Italic, Lucida Sans Bold, and Lucida Sans Bold Italic.

The following will be released soon: Lucida TEX Math Italic, Lucida TEX Math Symbol, Lucida TEX Math Extension, and Lucida Sans Typewriter. Dan notes that the Math fonts have the same character sets as shown in appendix F of *The TEXbook.* The Lucida Sans Typewriter has another union of Adobe's standard set, this time with the TEX text typewriter set, shown in figure 3 in the same appendix.

Bigelow suggested that for dates of release and information on TFM files, etc., one should contact Adobe. Once again, Dan Mills was most helpful. He said:

> As for TFM's, we've been getting a lot of help from Barry Smith of Kellerman & Smith ... to produce these. We aren't finished. If you are a TEXtures user, they will certainly be willing to help you out. If not, we plan to make these metrics available to the general TEX community 'somehow' in

[11] Another font with some similar design goals is Matthew Carter's Bitstream Charter.

the near future. Exactly how they will be distributed remains to be decided.

Imagen

Chuck Bigelow also noted that the Imagen Corporation currently offers various Lucida typefaces, and an upgrade to the full TEX character set will be released in July 1988. These will include the Lucida seriffed family, the Lucida Sans family, and the Lucida Sans Typewriter family. Imagen will produce the TEX Math fonts if there is demand from their users. Imagen Lucida will be available in outline format for their UltraScript (PostScript clone) and DDL language printers, and for their imPRESS printers. The Imagen fonts should be metrically compatible with the Adobe fonts. They are made from exactly the same outline data.

Compugraphic

A last minute update from Chuck indicates that Lucida will soon be available for Compugraphic typesetters:

> Compugraphic Corporation has licensed the Lucida seriffed family (roman, italic, bold, bold italic) and the three basic TEX math fonts (math italic, math symbol, math extension) for their 8600 and 9600 typesetters, and perhaps the 8400 as well. Contact Cynthia Marsh or Norbert Florendo at CG for estimate of availability, price, etc.

Compugraphic Corporation, Type Division, Wilmington, MA, (617) 658-5600.

Other Developments

A maker of inexpensive personal computers and printers will announce Lucida availability very shortly. The fonts will be metrically compatible with Adobe and Imagen.

Chuck and Kris are also working on outline versions of several of the additional fonts shown in Knuth's Volume E of *Computers and Typesetting*, and fonts analogous to the Euler family, including Bold Greek, Bold Script, Bold Symbol font, Small Capitals, as well as a Chancery, lower-case Script, Fraktur, Hebrew, and others. Light and Demibold versions of both seriffed and sans-serif families are also in progress, as well as the 'Bright' versions used in Scientific American. However, these must await a distributor like Adobe or Imagen to reach the market.

Chuck also reports that in their studio, Bigelow & Holmes, he and Kris have produced bitmap fonts in the TEX character set (as well as PostScript character set) in `PXL` format for the Lucida seriffed and Lucida Sans families in the following point sizes, at 300dpi: 6, 8, 10, 12, 14, 18, 24, and the same sizes, excepting 6 point, for 75 dpi and 100 dpi screens. Currently there is no distributor for these. These are hand-tuned bitmaps that are slightly different for each size, and therefore have their own (simple) `TFM` files that are not fully compatible with those for the Adobe PostScript outlines. However, they feel that the quality of the hand-tuned fonts is usually higher than that of the algorithmically produced bitmaps from outlines. They haven't finished bitmap versions of the math fonts or the typewriter fonts yet, but they do have an International Phonetic Alphabet and some other oddities in bitmap.

22.2 Contacts

Chuck Bigelow can be reached at the Dept. of Computer Science at Stanford:
Phone: (415) 723 3827
Arpa: `cab@sail.stanford.edu`,
or at his Menlo Park studio, Bigelow & Holmes:
Phone: (415) 326-8973.

Dan Mills can be reached at:
Adobe Systems Inc.,
1585 Charleston Road, P. O. Box 7900,
Mountain View, CA 94039-7900, USA.
Phone: (415) 962-2100
Net: `adobe!mills@decwrl`, (and possibly `mills@ucbvax.berkeley.edu`)

23 Icelandic

23.1 Pind

The Font

On March 10, 1988, Jorgen Pind reported in TEXhax 1988, issue 22, that he is running an unmodified TEX with new fonts and formats (including hyphenation) which cater for Icelandic.

Contact

Jorgen Pind,
Institute of Lexicography,
University of Iceland,
Reykjavik 101, Iceland.
Internet: `jorgen@lexis.hi.is`
Uucp: ...`mcvax!hafro!rhi!lexis!jorgen`

24 OCR-A

24.1 Lillqvist

In TEXhax 1987, issue 106, information was given about an OCR-A font coded in METAFONT84 by

Tor Lillqvist, VTT/ATK (Technical Research Centre of Finland, Computing Services).

The Font

Lillqvist's OCR–A is based on ISO Recommendation R1073, 1st ed., May 1969 (which he thinks is probably obsolete by now).

Terms of Availability

The font is distributed free of charge. On 1 June 1987, Brandon S. Allbery (`allbery@ncoast.UUCP`) and Michael Lichter posted the METAFONT sources for the OCR–A on Usenet, in `comp.sources.misc`.

Contact

Tor Lillqvist,
VTT/ATK,
Lehtisaarentie 2,
SF–00340 Helsinki, Finland.
Net: `tml@fingate.bitnet`, or
`tml@santra.UUCP`, or
`mcvax!santra!tml`

25 Miscellaneous

25.1 Austin Code Works

The Austin Code Works has a large collection of bitmap fonts that work with TEX, and which were originally created at SAIL (Stanford Artificial Intelligence Lab) in the late 60s or early 70s (I think). Because these fonts are not coded in up-to-date METAFONT, what you get is what you get, i.e., you cannot change the size or resolution of the fonts. I believe they are all (or most) 200dpi fonts. Bear in mind that although you might get, say, a Hebrew font, there are no accompanying macros to implement it at the keyboard level.

The Fonts

The 'KST Fonts by Les Earnest' are described thus in the ACW handout:

> Originally developed for the Xerox XGP printer, the 137 KST fonts include Hebrew, Greek, Old English, Old German, Cyrillic, hand [sign alphabet], and Tengwar alphabets in addition to the Roman alphabet in a large number of eclectic styles. Specify TEX or bitmap format. Both come with an extraction and display program.

The fonts include such essentials as single character fonts for the Stanford and MIT logos (separate fonts for each, naturally), two views of Snoopy, two views of Starship Enterprise, three fonts of chess pieces, several sans-serif fonts, and what looks as if it might be a very tiny Arabic font.

Terms of Availability

The collection of fonts costs $30.

Contact

The Austin Code Works,
11100 Leafwood Lane,
Austin, Texas 78750-3409, USA.
Phone: (512) 258-0785
BBS: (512) 258-8831
FidoNet: 1:382/12
Net: `acw!info@uunet.uu.net`

25.2 SPRITE.STY

If you use LATEX, and you only need one or two extra characters, an ingenious and very easy way to generate them has been devised by Martin Costabel. It is a LATEX style called `SPRITE`, and the code and documentation were published on 14 November 1987 in issue 1.8 of TEXMag, an online TEX magazine put out by Don Hosek (`dhosek@hmcvax.Bitnet`). Here is an extract from Martin's documentation:

> `SPRITE.STY` is a LATEX macro that allows you to define in a quick and dirty way your own symbols. You just have to define the character as a dot pattern on your screen and enclose it by `\sprite` and `\endsprite` commands. Of course, I know, TEX is awfully professional and this primitive technique will not provide results as good as a METAFONT–designed character or even one drawn using device-dependent `\special` commands, but if you just need one special character or some cute little symbol and you don't have the time/brains/Macintosh/superuser-privilege/money-for-AMS-fonts/or whatever-is-necessary for a professional solution, this might produce acceptable results.

Using `SPRITE.STY` one 'draws' the character to be defined as a pattern of characters on a grid. Figure 1 shows how *schwa* is done. To use this character in your LATEX document, all you have to do is use the command `\schwa`. This method uses a lot of TEX's memory, and is only suitable for characters which are used rarely, say a few times on a page.

Contact

Martin Costabel.
Net: `xbr1da29@ddathd21.bitnet`

```
\def\schwa{\FormOfSchwa\kern 1 pt}
    % Only necessary if \kern... is wanted
\sprite{\FormOfSchwa}(16,24)[0.4 em, 1 ex]
    % Resolution ca. 200x340 dpi.
:.......BBBBBBBBBB....... |
:....BBBB........BBBB.... |
:..BBB.............BBBB.. |
:.BB.................BBB. |
:.B...................BBB |
:.....................BBB |
:.....................BBB |
:.....................BBB |
:BBBBBBBBBBBBBBBBBBBBBBBB |
:BBB..................BBB |
:BBB..................BBB |
:BBB.................BBB. |
:.BBB...............BBB.. |
:..BBBB...........BBBB... |
:....BBBBB.....BBBBB..... |
:.......BBBBBBBB......... |
\endsprite
```

Figure 1: SPRITE.STY commands for *schwa.*

25.3 HP2TEX (HP Font Conversion)

The Program

In February 1988, the uuencoded ARC file HP2TEX appeared on Usenet, in comp.text. It included the Turbo Pascal 3.0 source code and a compiled DOS version of HP2TEX, a program to read a Hewlett Packard soft font and generate two files useful to TeX users, a PL and a PXL file.

Since the HP soft fonts are bitmap fonts generated from the Bitstream outline typefaces (see the **Bitstream** section), this is another way to produce the latter family of fonts. However, the TFM file produced by HP2TEX does not contain the information required to make ligatures, etc. (see below), so in practice, if real quality is sought, it would be better to buy the FIP from Personal TeX Inc.

Usage

The HP2TEX program prompts the user for the names of the font files for input and output, and also for the original design size of the HP font, and its magnification (so that a 12pt HP font can make a 12pt PXL font, or a 10pt PXL font magnified \magstep1). Names can be supplied on the command line, and will be given appropriate extensions if necessary. The output is a PXL font with its associated property list (PL) file. The PL file can be converted to a TFM file (with or without editing: see below) by the TeXware program PLtoTF (compiled DOS version available on the PCTeX Bulletin board: (415) 388-1708). The PXL file can be left as it is, or converted to a smaller PK file by PXtoPK (also on the same BBS).

The PL file generated by this program contains several parameters that determine the appearance of the TeX output. These parameters control the inter-word glue, the space after a period, the size of a quad and em space, and the parameters determining accent placement. The 'correct' values for these parameters are NOT contained in the HP soft fonts. They are estimated by HP2TEX, and might not be very good estimates. If your output is visually bad, read about fontdimen parameters in the back of the *TeXbook* (or METAFONT*book*) and adjust their values in the PL file. Then generate a new TFM file. The authors have not attempted to deal with kerns, although you could add those to the PL file as well. Ligatures require the same action, assuming the ligature glyph is in the font at all.

The authors note that certain fonts generated by GLYPHIX appear to convert with a bad underscore character that is far too low. This is in fact how the underscore is encoded in the font, not an artifact of the conversion.

Terms of Availability

HP2TEX was written by David Strip, with help from Dimitri Vulis. The program is copyrighted, and in keeping with the spirit of the TeX community, you are granted permission to copy and redistribute it so long as you provide the source and the README file along with any executable. In addition, you may not charge any fee in excess of the actual cost of the media and reasonable labor charges. This charge may not exceed $3.00 per disk plus shipping costs.

HP2TEX is available on SIMTEL20 (FTP or Listserve access) and GENIE, as well as some BBSs.

Contact

David Strip,
431 Camino de la Sierra NE,
Albuquerque, NM 87123, USA.
Arpa: drstrip@sandia-2.arpa
Uucp: ...(ucbvax, cmu!rice, ihnp4!lanl, gatech)!unmvax!intvax!drstrip

Dimitri L. Vulis,
529 W. 111 Street, #61
New York, NY 10025-1943, USA.
Bitnet: dlv@cunyvms1

25.4 Bar Codes

Issue 94 of TEXhax 1987 carried the following note from Dimitri Vulis:

> I was amazed to find out that business people pay enormous money for the ability to produce bar codes. This ought to be *trivial* with TEX. So, I got hold of a public domain BASIC program that supposedly does that and lifted the codes and put them into METAFONT (it was *easy*). Caveat(s): I have never tested these codes with an OCR. The BASIC program said it used '3 of 9' encoding. I presume it's not the same as UPC. The sizes may be off — I took them from the BASIC program that used HP LJ's 100dpi graphics mode.
>
> Remarks:
>
> - White space is a displayable character.
> - There are 9 significant strips, bwbwbwbwb; 3 of them are wide.
> - The `white_naro` at the end of each code is the inter-code spacing.
>
> The original BASIC program was written by 'Bill Wood Mil., WI' and later re-written by 'Bill Baines, Enfield, CT.'

Dimitri notes that he has not tested the bar codes.

Terms of Availability

The METAFONT code for the bar codes is available for anonymous FTP from `score.stanford.edu` as file `<TEX.TEXHAX>VULIS.TXH`. A copy has also been forwarded to TEX-L for BITNETers.

Contact

For Vulis's address see under **HP2TEX**.

25.5 Old English

Henderson

Doug Henderson has implemented METAFONT on the PC/AT family of personal computers, and is also responsible for generating the high resolution bitmaps of several of the non-standard fonts illustrated in recent issues of *TUGboat*, such as the APL and Greek fonts.

On Tuesday May 3, 1988, Doug wrote to me with several useful corrections to the present article, and included the following information:

> I have recently created a few characters for an associate at the International Christian University by the name of William Schipper. He requested that the Old English characters thorn (upper/lower case) and eth (upper/lower case) be created for him. As he was referred to me by Knuth, I decided to take this challenge and create the characters. Mainly, they are variations of characters found in Computer Modern already, with some polishing up.
>
> This will probably be the first in a series of ad-hoc characters I create for various folks in need, and I will keep them in a sort-of miscellaneous font category. For now, they are compatible with `CMR10`.

Doug is willing to distribute the Old English characters free of charge. He says that the final versions should be done in time for this year's TUG conference in Montréal. Contact him at:
Division of Library Automation,
Univ. of California, Berkeley,
186 University Hall,
Berkeley, CA 94720, USA.
Bitnet: `dlatex@ucbcmsa`
Arpa: `dlatex%ucbcmsa.cc@berkeley`

Curran

Charles Curran of the Oxford University Computing Service notes that he is 'generating odd squiggles for a user's medieval English', but he feels that the quality isn't adequate for wider dissemination at present. Contact Charles at:
OUCS,
13 Banbury Road,
Oxford, OX2 6NN, England.
Phone: (UK) 0865 56721
Net: `charles@vax.oxford.ac.uk`

Bradfield

In a note to TEXhax 1987, issue 73 (sent on Fri, 04 Sep 1987), Julian Bradfield reported that he has preliminary versions of the letters eth, thorn and yogh for Computer Modern. He said:

> They are rather hastily cobbled together from bits of real CM letters, so are not very robust at present; in particular my yogh is not yet very happy on lowres devices, though I hope to fix that soon. They look OK (to me!) in roman, italic, bold and sans-serif at 300 dpi, which is the highest resolution available to me.

If you want to use these characters, mail Julian at:
c/o Christ's College,
Cambridge, CB2 3BU, England.
Janet: `jcb7@uk.ac.cam.phx`
Earn/Bitnet: `jcb7@phx.cam.ac.uk`

Ridgeway and Barnett

A font to supplement the basic Latin alphabet of Computer Modern with Old English characters will be released in Autumn of this year by the University of Washington Humanities and Arts Computing Center. Macros for direct input and translators from various Personal Computer editor files will accompany the release.

For the contact address, see under **Tamil** above.

25.6 Vietnamese

A note from Trung Dung at the end of March 1988 signalled his intention of using METAFONT to create a Vietnamese font. This uses a Roman character set, with a wide range of diacritical marks. Trung hopes to finish some time this summer. He can be contacted at `trung@umb.edu`.

26 The PostScript Question

This article does not deal with the availability of PostScript fonts, because it would double its length to do so, and because the author has never used a PS device, and really wouldn't know what he was talking about. Suffice it to say that instead of sending a bitmap to the printer, a PostScript output file sends the printer instructions on how to construct character bitmaps at the time of printing. These PostScript bitmaps are created in a manner analogous to running METAFONT, in that the characters are encoded in a high level language, which gives the outline, filling or stroke routines for creating glyphs. Like TEX `TFM` files, PostScript fonts have their own font metric files, called `AFM` files. A utility program called `AFtoTF` exists to convert `AFM` files to `TFM` ones. A copy of `AFtoTF`, compiled and ready to run under DOS, is available on the Personal TEX BBS. Another route is to use `AFtoPL`, a program that converts the `AFM` file to the intermediate `PL` format, which can then be converted to a `TFM` file (see **HP2TEX** above). The C source for the latest version of `AFtoPL` (version 2.0, 25 April 1988) by Clayton M. Elwell is available via anonymous FTP from `tut.cis.ohio-state.edu` as `pub/aftopl.c`, and also from `june.cs.washington.edu`. Since all the font information TEX needs to create a `DVI` file is in the `TFM` file, TEX can thus create `DVI` files ready to be printed using PostScript fonts. All that is needed is a `DVI` driver that knows how to call for a PostScript font in the right way, and such drivers are available. Thus, with a bit of tinkering, it is perfectly feasible to use PostScript fonts in a TEX document, and there is a large and growing catalogue of such fonts. Adobe themselves have a catalogue of fonts which they market directly, which currently includes 230 individual fonts (several for most typefaces). There are also several other companies and individuals producing PostScript fonts in various styles and for a wide range of languages, using such font creation tools as Fontographer on the Mac, or Publisher's Type Foundry on the PC.

Output Devices

Index to Sample Output from Various Devices

Camera copy for this issue of TUGboat was prepared on the devices indicated, and can be taken as representative of the output produced by those devices. The bulk of this issue was at the American Mathematical Society on a VAX 8600 (VMS) and output on an APS-μ5 using resident CM fonts and additional downloadable fonts for special purposes. The items listed below were received as camera copy; they were prepared on the devices indicated. The output devices used to prepare the advertisements were not usually identified; anyone interested in determining the device used for a particular ad should inquire of the advertiser.

- Unidentified: all advertisements. Some of the ads were received in a size larger than permitted; these were reduced photographically using the PMT process.
- Autologic APS-μ5 (1440 dpi): Donald E. Knuth, *A punk Meta-Font*, p. 152, DEC 10; the drawings and proof fonts on pp. 154–156 and the photograph on p. 155 were pasted in.
- Canon CX (300 dpi): Georgia Tobin, *Designing for low-res devices*, p. 126.

A Punk Meta-Font

Donald E. Knuth
Stanford University

In February, 1985, Gerard and Marjan Unger gave a series of nine evening lectures at Stanford, in which they surveyed the evolution of styles in art, architecture, clothing, product design, and typography during the past 75 years. The lectures were especially interesting because they revealed the way in which changes in typographic fashions were juxtaposed with the changes in other kinds of fashions. The Ungers demonstrated a remarkable fact, that *typography tends to lag behind other stylistic changes by about ten years.*

When I woke up on the morning of their final lecture, I suddenly realized that there was an obvious corollary of what they had been saying during the previous eight evenings: It was now about time to design a typeface based on trends that had emerged during the late 70s! Furthermore, I also had a reasonably clear idea of what such a design might be like, because the lectures had turned up a strong similarity between some "punk" graphics exhibited in London and a certain lines-and-dots motif found in the upholstery of some "punk" furniture designed in Italy.

A lines-and-dots motif is trivially easy for METAFONT to handle, so I decided to create a new family of typefaces called PUNK. I spent several pleasant hours at the computer terminal that afternoon; and by evening I was able to present everybody in the audience with an up-to-the-minute souvenir of the Unger's lectures, laserprinted in PUNK10 and PUNK20.

The idea of PUNK was to start with more-or-less traditional stick-letter shapes, but to ask METAFONT to perturb the key points by random amounts so that the letters look a bit deranged. Here, for example, are several texts set with a few varieties of PUNK fonts:

ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789 ÆŒØΓΔΘΛΞ ΠΣΥΦΨΩ æœøßıȷ`´ˇ˘¯˚¸^~¨˝˙ ¸˛ .,;:¿?¡!‘’“”#$%&@-–—* ()[]↑↓+/<=>

PUNK10

‘Humph!’ said Artie. His face was red and he was certainly flustered. ‘It ’d be a dead lucky thing if some more people around the shop ’d change a little. They could n’t be any punker ’n they are now.’ [1]

ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789 ÆŒØΓΔΘΛΞΠΣΥΦΨΩ æœøßıȷ`´ˇ˘¯˚¸^~¨˝˙ ¸˛ .,;:¿?¡!‘’“”#$%&@-–—* ()[]↑↓+/<=>

PUNK12

A forked vein began to swell in Spade’s forehead. ... His voice became persuasive again. ‘Listen, Gutman, we’ve absolutely got to give them a victim. There’s no way out of it. Let’s give them the punk.’ He nodded pleasantly at the boy in the doorway. [3]

ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz 0123456789 ÆŒØΓΔΘΛΞΠΣΥΦΨΩ æœø ßıȷ`´ˇ˘¯˚¸^~¨˝˙ ¸˛ .,;:¿?¡!‘’“”#$%&@-–—* ()[] ↑↓+/<=>

PUNK20

Pistol. [*Aside.*] This punk is one of Cupid’s carriers. Clap on more sails; pursue; up

with your fights; Give fire! She is my prize,
or ocean whelm them all! [*Exit.*] [6]

ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmn opqrstuvwxyz 0123456789 ÆŒØΓΔΘΛΞΠΣΥΦΨΩ æœø ßıȷ ` ´ ˇ ˘ ¯ ˚ ¸ ^ ˜ ¨ ˝ ˙ ¸ ˛ .,;:¿?¡!“”‘’#$%&@-–— ()[] ↑↓+/<=>*

PUNKSL20

Punk rock is the generic term for the latest musical garbage bred by our troubled culture, British and American.

Johnny Rotten and the Sex Pistols are punks. They sing ‘Anarchy in the UK,’ which ends with a scream: ‘Destroy.’ Clash and Damned are other bands.

Punk will fade. Its apologists are ludicrous. There are ways to protest about the putrid faces of both pop and society without relapsing into barbarism. Punk is anti-life, anti-humanity. [2]

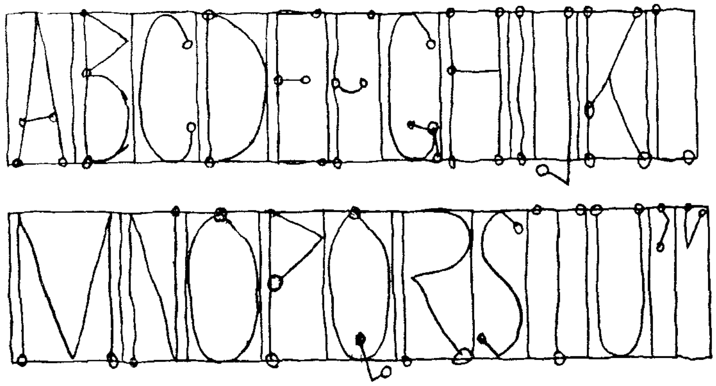
ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdef ghijklmnopqrstuvwxyz 0123456789 ÆŒØ ΓΔΘΛΞΠΣΥΦΨΩ æœøßıȷ ` ´ ˇ ˘ ¯ ˚ ¸ ^ ˜ ¨ ˝ ˙ ¸ ˛ .,;:¿?¡!“”‘’#$%&@-–—* ()[]↑↓+/<=>

PUNKBX20

Generation of letterforms by mathematical means was first tried in the fifteenth century; it became

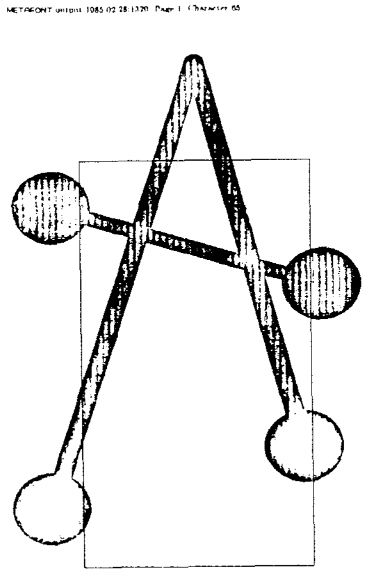
POPULAR IN THE SIXTEENTH AND SEVENTEENTH CENTURIES; AND IT WAS ABANDONED (FOR GOOD REASONS) DURING THE EIGHTEENTH CENTURY. [4]

At the time I "designed" these typefaces, I had just begun to make the final version of the Computer Modern fonts by converting my Almost Computer Modern code to the conventions of METAFONT84. The letters 'A', 'B', 'C', and 'D' had been debugged so far, and I was planning to tackle 'E' soon; but I felt like taking a break. So I made sketches of some punkish forms, as follows:



(I used a large sheet of graph paper; this illustration has been reduced to about 1/4 of the original size.)

At 1 p.m. I went to the computer and began to compose a simple base file. Not much had to be done, since plain METAFONT already includes most of the basic routines; so I had my first proof output at 1:20 p.m.:

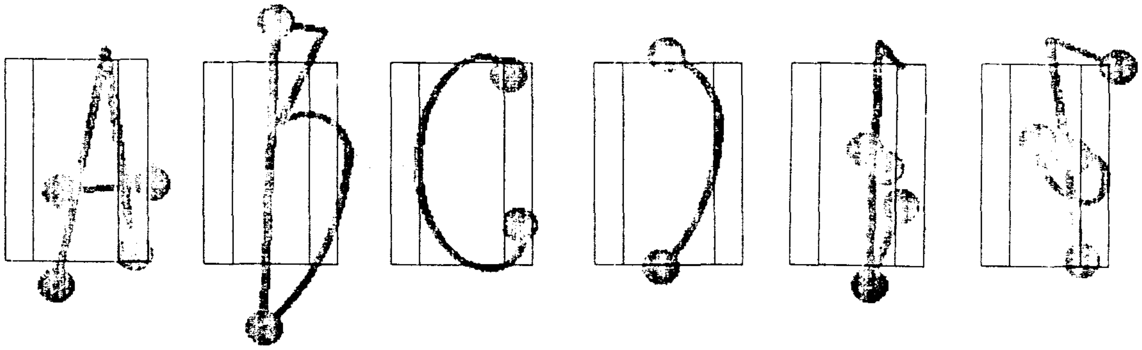


(40% of original size)

The letter 'A' seemed to be working, so I proceeded to type the METAFONT programs for 'B' through 'Z'. I decided to type everything before looking at any proofs, so I simply translated the sketches into METAFONT constructions, composing everything at the keyboard. It wasn't necessary to make accurate measurements, because random perturbations to the points were going to be made anyway; so I soon got used to the conventions of this font, and I was limited only by typing speed. I didn't even need sketches of the letters 'V' through 'Z', because it was easy to imagine what they would be in algebraic form. At 3:04 p.m. the typing was done, and I was able to run METAFONT and get proofs of all 26 uppercase letters. I also had thrown in a few punctuation marks (period, comma, opening and closing single and double quotation marks).

Of course there were bugs in my code. For example, the first few letters came out looking like this:

METAFONT output 1985.02.28:1504



(I refuse to show you the first form of the letter 'G'.) But by 4 p.m. I was ready to make the first trial setting of text:

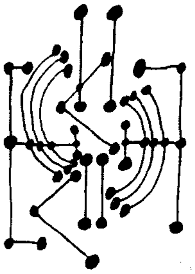
THIS 'PUNK' ALPHABET, INSPIRED BY MARJAN'S LECTURE LAST NIGHT, WAS DESIGNED BY METAFONT'S RANDOM NUMBER GENERATOR.

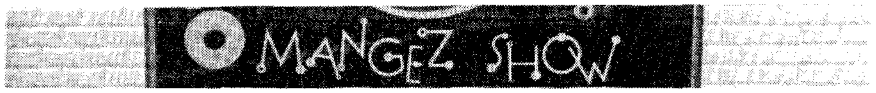
THE QUICK BROWN FOX JUMPED OVER THE LAZY HAMBURGEFONS.

At this point an unexpected glitch slowed things down a bit: The letters of this font had some unusual characteristics that hadn't arisen in GF files before, so a bug showed up in our METAFONT-to-laserprinter software. I made copies of the offending files, for later reference, and I was able to get around the bug by choosing another random seed and generating the font again. After another half hour of tuning things up (and toning down the randomness a bit), I was able to go home for supper.

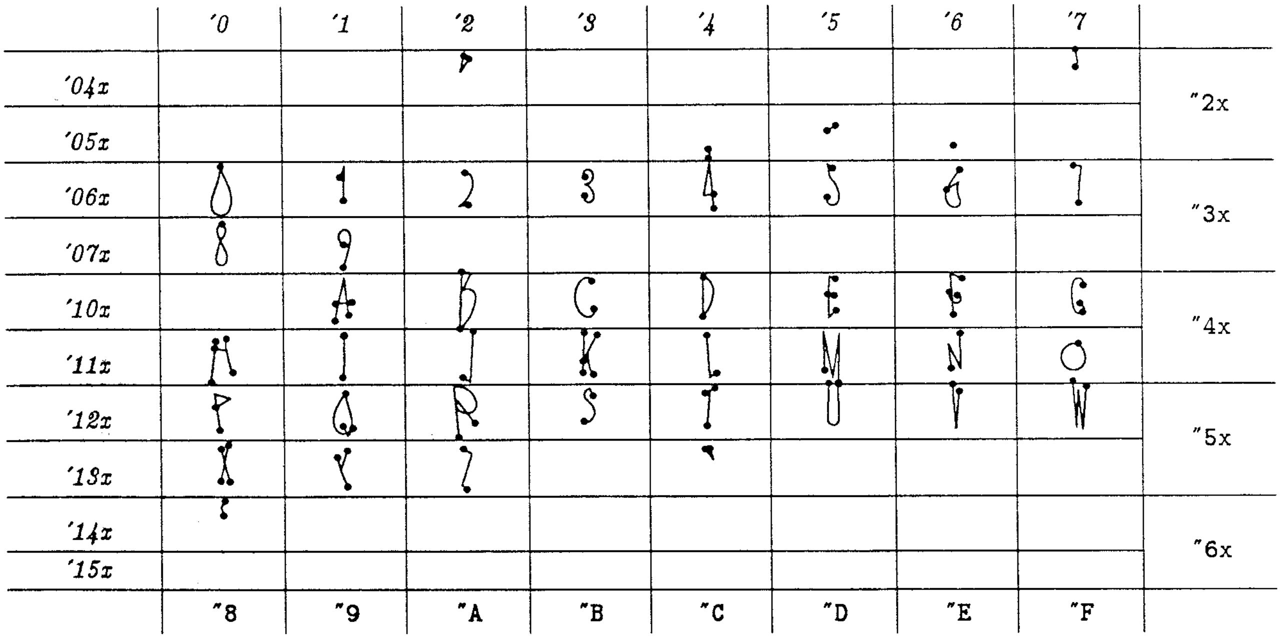
During the supper hour, I realized that a proper keepsake for that evening would include the typeset date. So I gulped down my meal, quickly sketched a set of numerals, and raced back to my office. Soon I had the font of 43 characters shown at the top of the next page. Whew! I was ready to hand out a sample sheet to everybody at 7 p.m., hot off the copy machine.

A year or so later, I was wondering around in Boston's Museum of Fine Arts and I came across a drawing made by Picasso in 1924 [5]. (See the illustration at the right.) This made me wonder if the PUNK fonts weren't really sixty years behind the times, not just ten. On the other hand, I found a striking confirmation of the relevance of at least part of the PUNK design in October, 1986, when I chanced to see the following typography on a billboard in the Paris Métro(!):





Test of punk20 on February 28, 1985 at 1824


'0
'1
'2
'3
'4
'5
'6
'7
'04x
'05x
'06x
'07x
'10x
'11x
'12x
'13x
'14x
'15x
"2x
"3x
"4x
"5x
"6x
"8
"9
"A
"B
"C
"D
"E
"F

In February, 1987, I decided to extend this original font to the full TEX character set. The extra programming didn't take long, since I decided to generate the lowercase letters as "small caps," and since each new character could be typed into the computer in one or two minutes. About one third of the characters had to be revised after I saw proofsheets, since they looked either too punk or not punk enough; and one-third of the revised characters had to be revised again; and so on. But after about six hours of additional work, a complete PUNK meta-font with 128 characters was ready for use (in case anybody wanted it).

In the remainder of this paper, I'll present the details of the METAFONT code, since this may be the shortest possible example of METAFONT programs for a family of complete 128-character fonts.

The programs appear in several different kinds of files, as explained in Appendix E of [4]: There are parameter files, to specify specific fonts of the family; there is a driver file, which controls most of the font generation process; and there are program files, which contain the code for individual characters. (I didn't need a base file, since the special macros for these fonts could all be included in the driver file.)

Here is a typical parameter file, PUNK20.MF:

```
% 20-point PUNK font:
designsize := 20pt#; font_identifier := "PUNK";
ht# := 14pt#;                          % height of characters
u# := 4/9pt#;                          % unit width
s# := 2pt#;                            % extra sidebar
px# := .8pt#;                          % horizontal thickness of pen
py# := .6pt#;                          % vertical thickness of pen
dot# := 2.7pt#;                        % diameter of dots
dev# := .5pt#;                         % standard deviation of punk points
slant := 0;                            % obliqueness
```

```
seed := 2.71828;                                         % seed for random number generator
input PUNK                                               % switch to the driver file
```

Its purpose is to customize the meta-design to a particular selection of sizes and weights. The parameters used to define the five fonts exhibited earlier in this article are:

	PUNK10	PUNK12	PUNK20	PUNKSL20	PUNKBX20
designsize	$10pt^{\#}$	$12pt^{\#}$	$20pt^{\#}$	$20pt^{\#}$	$20pt^{\#}$
font_identifier	"PUNK"	"PUNK"	"PUNK"	"PUNKSL"	"PUNKBX"
$ht^{\#}$	$7pt^{\#}$	$8.4pt^{\#}$	$14pt^{\#}$	$14pt^{\#}$	$14pt^{\#}$
$u^{\#}$	$1/4pt^{\#}$	$.3pt^{\#}$	$4/9pt^{\#}$	$4/9pt^{\#}$	$.6pt^{\#}$
$s^{\#}$	$1.2pt^{\#}$	$1.4pt^{\#}$	$2pt^{\#}$	$2pt^{\#}$	$2.2pt^{\#}$
$px^{\#}$	$.6pt^{\#}$	$.75pt^{\#}$	$.8pt^{\#}$	$.8pt^{\#}$	$2pt^{\#}$
$py^{\#}$	$.5pt^{\#}$	$.62pt^{\#}$	$.6pt^{\#}$	$.6pt^{\#}$	$1.6pt^{\#}$
$dot^{\#}$	$1.3pt^{\#}$	$1.6pt^{\#}$	$2.7pt^{\#}$	$2.7pt^{\#}$	$3.5pt^{\#}$
$dev^{\#}$	$.3pt^{\#}$	$.36pt^{\#}$	$.5pt^{\#}$	$.5pt^{\#}$	$.5pt^{\#}$
slant	0	0	0	1/3	0
seed	sqrt 2	sqrt 3	2.71828	3.14159	0.57722

The driver file PUNK.MF was the most difficult to write, because it contains the "essence" of the design. The various parts of this file grew one step at a time. For example, the last two parameters of the '**beginpunkchar**' macro were added after I noticed that some characters can't tolerate as much random deviation in their points as a normal character can (otherwise they become unrecognizable).

```
% This is PUNK, a meta-font inspired by Gerard and Marjan Unger's lectures, February 1985
mode_setup;
randomseed := seed;
define_pixels(u, dev);
define_blacker_pixels(px, py, dot);
define_whole_pixels(s);
xoffset := s;
pickup pencircle xscaled px yscaled py; punk_pen := savepen;
pickup pencircle scaled dot; def_pen_path_;
path dot_pen_path; dot_pen_path := currentpen_path;
currenttransform := identity slanted slant yscaled aspect_ratio;
def beginpunkchar(expr c, n, h, v) =                                % code c; width is n units
   hdev := h * dev; vdev := v * dev;           % modify horizontal and vertical amounts of deviation
   beginchar(c, n * u#, ht#, 0); italcorr ht# * slant; pickup punk_pen enddef;
extra_endchar := extra_endchar & "w:=w+2s;charwd:=charwd+2s#";
def ↑ = transformed currenttransform enddef;
def makebox(text rule) =
  for y = 0, h:
   rule((-s, y)↑, (w - s, y)↑); endfor                                        % horizontals
  for x = -s, 0, w - 2s, w - s: rule((x, 0)↑, (x, h)↑); endfor                  % verticals
  enddef;
rulepen := pensquare;
vardef pp expr z = z + (hdev * normaldeviate, vdev * normaldeviate) enddef;
def pd expr z = addto_currentpicture contour
   dot_pen_path shifted z_t_ withpen penspeck enddef;                           % drawdot
input PUNKL                                                           % uppercase letters
input PUNKAE                                                          % uppercase Æ, Œ, Ø
input PUNKG                                                           % uppercase greek
input PUNKP                                                   % punctuation
input PUNKD                                                   % digits
input PUNKA                                                   % accents
ht# := .6ht#;  dev := .7dev;
input PUNKSL                                                  % special lowercase
extra_beginchar := extra_beginchar & "charcode:=charcode+32;";
input PUNKL                                                   % lowercase letters
extra_beginchar := extra_beginchar & "charcode:=charcode-35;";
input PUNKAE                                                  % lowercase æ, œ, ø
font_slant := slant;
font_quad := 18u# + 2s#;
font_normal_space := 9u# + 2s#;
font_normal_stretch := 6u#;
font_normal_shrink := 4u#;
font_x_height := ht#;
font_coding_scheme := "TeX text without f-ligatures";
end
```

The 128 characters generated by PUNK.MF have the same font positions as the characters in fonts like cmr5 and cmcsc10 that don't have f-ligatures. Here, for example, is the layout of the font PUNKZ20, which is like PUNK20 except that *dev* = 0 (so that there is no randomness):

	´0	´1	´2	´3	´4	´5	´6	´7	
´00x	Γ	Δ	Θ	Λ	Ξ	Π	Σ	Υ	″0x
´01x	Φ	Ψ	Ω	↑	↓	'	¡	¿	
´02x	ı	ȷ	`	´	ˇ	˘	¯	˚	″1x
´03x	¸	ß	æ	œ	ø	Æ	Œ	Ø	
´04x	-	!	”	#	$	%	&	’	″2x
´05x	(	)	*	+	,	-	.	/	
´06x	0	1	2	3	4	5	6	7	″3x
´07x	8	9	:	;	<	=	>	?	
´10x	@	A	B	C	D	E	F	G	″4x
´11x	H	I	J	K	L	M	N	O	
´12x	P	Q	R	S	T	U	V	W	″5x
´13x	X	Y	Z	[	“	]	ˆ	˙	
´14x	‘	a	b	c	d	e	f	g	″6x
´15x	h	i	j	k	l	m	n	o	
´16x	p	q	r	s	t	u	v	w	″7x
´17x	x	y	z	–	—	˝	˜	¨	
	″8	″9	″A	″B	″C	″D	″E	″F	

Let's look now at the program files. The first one I wrote was PUNKL.MF, which defines all the letters from A to Z:

```
% Punk letters:

beginpunkchar("A", 13, 1, 2);
z1 = pp(1.5u, 0); z2 = (.5w, 1.1h); z3 = pp(w - 1.5u, 0);
pd z1; pd z3; draw z1 -- z2 -- z3;                                % left and right diagonals
z4 = pp.3[z1, z2]; z5 = pp.3[z3, z2]; pd z4; pd z5; draw z4 -- z5;                  % crossbar
endchar;

beginpunkchar("B", 12, 1, 1);
z1 = pp(2u, 0); z2 = pp(2u, .6h); z3 = pp(2u, h); pd z1; pd z3; draw z1 -- z3;          % stem
z1.5 = pp(w - u, .5y2); z2.5 = pp(w - u, .5[y2, y3]); draw z2 -- z2.5 -- z3;      % upper lobe
draw flex(z2, z1.5, z1);                                                          % lower lobe
endchar;

beginpunkchar("C", 13, 1, 2);
z1 = pp(w - 2u, .8h); z2 = pp(.6w, h); z3 = pp(u, .5h); z4 = (.6w, 0); z5 = (w - 2u, .2h);
pd z1; pd z5; draw z1 .. z2 .. z3 .. z4 .. z5;                                           % arc
endchar;

beginpunkchar("D", 14, 1, 2);
z1 = pp(2u, 0); z2 = pp(2u, h); z3 = pp(w - u, .6h); pd z1; pd z2; draw flex(z1, z3, z2);  % lobe
draw z1 -- z2;                                                                          % stem
endchar;

beginpunkchar("E", 12, .5, 1);
z1 = pp(2u, 0); z2 = pp(2u, h); z3 = pp(w - 2.5u, h); z4 = pp(w - 2u, 0);
pd z3; pd z4; draw z4 -- z1 -- z2 -- z3;                                      % stem and arms
z5 = pp(2u, .6h); z6 = pp(w - 3u, .6h); pd z5; pd z6; draw z5 -- z6;                % crossbar
endchar;

beginpunkchar("F", 12, .5, 2);
z1 = pp(2u, 0); z2 = pp(2u, h); z3 = pp(w - 2u, h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                             % stem and arm
z5 = pp(2u, .6h); z6 = pp(w - 3u, .6h); z4 = pp.5[z5, z6] - (0, .1h);
pd z5; pd z6; draw flex(z5, z4, z6);                                                % crossbar
endchar;

beginpunkchar("G", 13, .5, .5);
z1 = pp(w - 2u, .8h); z2 = pp(.6w, h); z3 = pp(u, .5h); z4 = pp(.6w, 0); z5 = (w - 2u, 0);
pd z1; draw z1 .. z2 .. z3 .. z4 --- z5;                                                 % arc
z6 = pp(.5[u, x5], .4h); pd z6; pd z5; draw z6 -- (pp(x5, y6)) -- z5;                   % spur
endchar;

beginpunkchar("H", 14, 1, .5);
z1 = pp(2u, 0); z2 = pp(2u, h); z3 = pp(w - 2u, 0); z4 = pp(w - 2u, h);
z5 = pp(2u, .6h); z6 = pp(w - 2u, .6h);
pd z1; pd z2; pd z3; pd z4; draw z1 -- z2; draw flex(z3, z6, z4);                      % stems
pd z5; draw z5 -- z6;                                                               % crossbar
endchar;

beginpunkchar("I", 5, 1, 2);
z1 = pp(.5w, 0); z2 = (.5w, 1/3h); z3 = (.5w, 2/3h); z4 = (.5w, h);
pd z1; pd z4; draw flex(z1, z2, z3, z4);                                                % stem
endchar;

beginpunkchar("J", 9, 1, 2);
z1 = pp(w - 2u, h); z2 = pp(w - 2u, -.1h); z3 = pp(u, 0); pd z1; pd z3; draw z1 -- z2 -- z3;  % arc
endchar;
```

```
beginpunkchar("K", 14, 1, 2);
z1 = pp(2u, 0);  z2 = pp(2u, h);  z3 = pp(2u, 1/3h);  z4 = pp(w - 1.5u, h);
pd z1;  pd z2;  draw z1 -- z2;                                                    % stem
pd z3;  pd z4;  draw z3 -- z4;                                          % upper diagonal
z6 = pp(w - u, 0);  z5 = 1/3[z3, z4];  pd z6;  draw flex(z5, .8[z1, 2/3[z5, z6]], z6);  % lower diagonal
endchar;
beginpunkchar("L", 11, 1, 2);
z1 = pp(2u, h);  z2 = pp(2u, 0);  z3 = pp(w - 1.5u, 0);
pd z1;  pd z3;  draw z1 -- z2 -- z3;                                      % stem and arm
endchar;
beginpunkchar("M", 17, .5, 2);
z1 = pp(2u, 0);  z2 = pp(2u, h);  z3 = pp(.5w, 0);  z4 = pp(w - 2u, h);  z5 = pp(w - 2u, 0);
pd z1;  pd z5;  draw z1 -- z2 -- z3 -- z4 -- z5;                % stems and diagonals
endchar;
beginpunkchar("N", 13, .75, 2);
z1 = pp(2u, 0);  z2 = pp(2u, h);  z3 = pp(w - 2u, 0);  z4 = pp(w - 2u, h);
pd z1;  pd z4;  draw z1 -- z2 -- z3 -- z4;                      % stems and diagonals
endchar;
beginpunkchar("O", 12, .5, 2);
z1 = pp(.5w, h);  z2 = pp(u, .55h);  z3 = pp(.5w, 0);  z4 = pp(w - u, .55h);
pd z1;  draw z1{left} .. z2 .. z3 .. z4 .. z1;                                    % bowl
endchar;
beginpunkchar("P", 13, 1, 2);
z1 = pp(2u, 0);  z2 = pp(2u, 1.1h);  z3 = pp(2u, .5h);  z4 = pp(w, .6[y3, y2]);
pd z1;  pd z3;  draw z1 -- z2 -- z4 -- z3;                               % stem and bowl
endchar;
beginpunkchar("Q", 14, .5, 2);
z1 = pp(.5w, h);  z2 = pp(u, .55h);  z3 = pp(.5w, 0);  z4 = pp(w - u, .55h);
pd z1;  draw z1{curl 2} .. z2 .. z3 .. z4 .. z1;                                  % bowl
z5 = pp(.4w, .2h);  z6 = pp(w - u, -.1h);  z7 = pp(.5[x5, x6], -.2h);
pd z5;  pd z6;  draw z5 -- z7 -- z6;                                              % tail
endchar;
beginpunkchar("R", 16, 1, 2);
z1 = pp(2u, 0);  z2 = pp(2u, h);  z3 = pp(w - u, .6[y2, y4]);  z4 = pp(2u, .5h);  z5 = pp(w - 1.5u, 0);
pd z1;  pd z2;  pd z5;  draw z1 -- flex(z2, z3, z4) -- z5;     % stem, bowl, and diagonal
endchar;
beginpunkchar("S", 11, .3, 1);
z1 = pp(w - 2u, .9h);  z2 = pp(.5w, h);  z3 = pp(u, .7h);  z4 = .6[z6, z2];
z5 = pp(w - u, .35h);  z6 = pp(.5w, u);  z7 = pp(u, .2h);
pd z1;  pd z7;  draw z1 -- z2 ... z3 .. z4 .. z5 ... z6 -- z7;                   % stroke
endchar;
beginpunkchar("T", 13, .75, 2);
z1 = pp(u, h);  z2 = pp(w - u, h);  z3 = pp(.5w, 0);
pd z1;  pd z2;  pd z3;  draw z1 -- z2;                                            % arms
draw .5[z1, z2] -- z3;                                                            % stem
endchar;
beginpunkchar("U", 13, .3, 2);
z1 = pp(2u, h);  z2 = pp(2u, .2h);  z3 = pp(.5w, 0);  z4 = pp(w - 2u, .2h);  z5 = pp(w - 2u, h);
pd z1;  pd z5;  draw z1 --- z2 ... z3{z4 - z2} ... z4 --- z5;                    % stroke
endchar;
```

```
beginpunkchar("V", 13, 1, 2);
z1 = pp(1.5u, h); z2 = pp(.5w, 0); z3 = pp(w - 1.5u, h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                        % diagonals
endchar;

beginpunkchar("W", 18, 1, 2);
z1 = pp(1.5u, h); z2 = pp(.5[x1, x3], 0); z3 = pp(.5w, .8h); z4 = pp(.5[x3, x5], 0); z5 = pp(w - 1.5u, h);
pd z1; pd z5; draw z1 -- z2 -- z3 -- z4 -- z5;                            % diagonals
endchar;

beginpunkchar("X", 13, 1, 1);
z1 = pp(1.5u, h); z2 = pp(w - 1.5u, 0); z3 = pp(1.5u, 0); z4 = pp(w - 2.5u, h);
pd z1; pd z2; draw z1 -- z2;                                              % main diagonal
pd z3; pd z4; draw z3 -- z4;                                              % cross diagonal
endchar;

beginpunkchar("Y", 13, 1, 2);
z1 = pp(1.5u, h); z2 = pp(w - 1.5u, h); z3 = pp(.5w, .5h); z4 = pp(.5w, 0);
pd z1; pd z2; pd z4; draw z1 -- z3 -- z4;                                 % stem and left diagonal
draw z2 -- z3;                                                            % right diagonal
endchar;

beginpunkchar("Z", 11, 1, 2);
z1 = pp(1.5u, h); z2 = pp(w - 2.5u, h); z3 = pp(1.5u, 0); z4 = pp(w - 1.5u, 0);
pd z1; pd z4; draw z1 -- z2 -- z3 -- z4;                                  % diagonals
endchar;
```

(It slowed me down a little to type the comments that identify the strokes. But such comments are enormously valuable when characters are being revised, so I knew that I should include them right from the beginning.)

Three of the letters go into a special file, PUNKAE.MF, because the character codes of these uppercase letters have a nonstandard relation to the character codes of the corresponding lowercase equivalents:

```
beginpunkchar(oct "035", 16, 1, 2);                                       % Æ
z1 = pp(1.5u, 0); z2 = pp(.6w, h); z3 = pp(w - 1.5u, h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                        % left diagonal and upper arm
z4 = pp.3[z1, z2]; z5 = pp(.6w, 0); z6 = pp(w - 2u, .3h); pd z4; pd z6; draw z4 -- z6;   % crossbar
z7 = pp(w - u, 0); pd z2; pd z7; draw z2 -- z5 -- z7;                     % stem and lower arm
endchar;

beginpunkchar(oct "036", 18, 1, 2);                                       % Œ
z1 = pp(.5w, h); z2 = pp(u, .4h); z3 = pp(.5w, 0); pd z1; draw z1 .. z2 .. {right}z3;   % bowl
z4 = pp(w - 1.5u, h); z5 = pp(w - 2u, .4h); z6 = pp(w - u, 0);
pd z4; pd z6; draw z4 -- z1 -- z3 -- z6;                                  % arms and stem
pd z5; draw z5 -- .4[z3, z1];                                             % crossbar
endchar;

beginpunkchar(oct "037", 14, 1, 1);                                       % Ø
z1 = pp(.5w, h); z2 = pp(u, .5h); z3 = pp(.5w, 0); z4 = pp(w - u, .5h);
z5 = pp(w - 2u, 1.1h); z6 = pp(2u, -.1h);
pd z1; pd z6; draw z1 .. z2 .. z3 .. z4 .. z5 -- z6;                      % bowl and diagonal
endchar;
```

There's also a special file PUNKSL.MF for lowercase letters with no matching uppercase:

```
beginpunkchar(oct "020", 5, 1, 2);                                        % dotless I
z1 = pp(.5w, 0); z2 = (.5w, 1/3h); z3 = (.5w, 2/3h); z4 = (.5w, h);
pd z1; pd z4; draw flex(z1, z2, z3, z4);                                  % stem
endchar;
```

```
beginpunkchar(oct "021", 9, 1, 2);                                                  % dotless J
z1 = pp(w − 2u, h); z2 = pp(w − 2u, −.1h); z3 = pp(u, 0); pd z1; pd z3; draw z1 -- z2 -- z3;   % arc
endchar;

beginpunkchar(oct "031", 18, .3, 1);                                                % German sharp S
z1 = pp(.5w − u, .9h); z2 = pp(1/3w, h); z3 = pp(u, .7h); z4 = .6[z6, z2];
z5 = pp(.5w, .35h); z6 = pp(1/3w, u); z7 = pp(u, .2h);
pd z1; pd z7; draw z1 -- z2 ... z3 .. z4 .. z5 ... z6 -- z7;                        % left stroke
for i = 1 upto 7: z[i + 10] = pp(z[i] shifted (.5w − u, 0)); endfor
pd z11; pd z17; draw z11 -- z12 ... z13 .. z14 .. z15 ... z16 -- z17;               % right stroke
endchar;
```

The uppercase Greek letters in file PUNKG.MF may have a slightly different style than those of PUNKL, because I wrote them two years later. Is there an obvious difference?

```
beginpunkchar(oct "000", 11, 1, 2);                                                 % Γ
z1 = pp(2u, 0); z2 = pp(2u, h); z3 = pp(w − 1.5u, h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                                  % stem and arm
endchar;

beginpunkchar(oct "001", 15, 1, 2);                                                 % Δ
z1 = pp(u, 0); z2 = pp(.5w, h); z3 = pp(w − u, 0);
pd z1; draw z1 -- z2 .. tension 5 .. z3 .. tension 5 .. z1;                         % triangle
endchar;

beginpunkchar(oct "002", 15, .5, 2);                                                % Θ
z1 = pp(.5w, h); z2 = pp(u, .6h); z3 = pp(.5w, 0); z4 = pp(w − u, .6h);
pd z1; draw z1 .. tension .8 .. z2 .. z3 .. z4 .. tension .8 .. z1;                 % bowl
z5 = pp(x2 + 2u, .4h); z6 = pp(x4 − 2u, .4h); pd z5; pd z6; draw z5 -- z6;          % bar
endchar;

beginpunkchar(oct "003", 12, 1, 2);                                                 % Λ
z1 = pp(u, 0); z2 = pp(.5w, h); z3 = pp(w − u, 0);
pd z1; pd z3; draw z1 -- z2 -- z3;                                                  % diagonals
endchar;

beginpunkchar(oct "004", 12, 1, 1);                                                 % Ξ
z1 = pp(u, h); z2 = pp(w − u, h); pd z1; pd z2; draw z1 -- z2;                      % upper arm
z3 = pp(2u, .55h); z4 = pp(w − 2u, .55h); pd z3; pd z4; draw z3 -- z4;              % bar
z5 = pp(u, 0); z6 = pp(w − u, 0); pd z5; pd z6; draw z5 -- z6;                      % lower arm
endchar;

beginpunkchar(oct "005", 13, 1, .5);                                                % Π
z1 = pp(1.5u, 0); z2 = pp(1.5u, h); z3 = pp(w − 1.5u, h); z4 = pp(w − 1.5u, 0);
pd z1; pd z4; draw z1 -- z2 -- z3 -- z4;                                            % stems and bar
endchar;

beginpunkchar(oct "006", 13, 1, 1);                                                 % Σ
z1 = pp(w − u, h); z2 = pp(u, h); z3 = pp(.5w − u, .5h); z4 = pp(u, 0); z5 = pp(w − u, 0);
pd z1; pd z5; draw z1 -- z2{.5[z4, z5] − z2} .. z3 -- z4 -- z5;                     % arms and diagonals
endchar;

beginpunkchar(oct "007", 15, 1, .5);                                                % Υ
z1 = pp(u, .8h); z2 = pp(.3w, h); z3 = pp(.5w, .5h); z4 = pp(.5w, 0);
pd z1; pd z4; draw z1 .. z2 .. tension 2 .. z3 --- z4;                             % left arc and stem
z5 = pp(w − u, .8h); z6 = pp(.7w, h); pd z5; draw z5 .. z6 .. tension 2 .. {z4 − z3}z3;   % right arc
endchar;
```

```
beginpunkchar(oct "010", 13, 1, 2);                                              % Φ
z1 = pp(.5w, h); z2 = pp(.5w, 0); pd z1; pd z2; draw z1 -- z2;                   % stem
z3 = pp(.5w, 2/3h); z4 = pp(u, .5h); z5 = pp(.5w, 1/4h); z6 = pp(w - u, .5h);
pd z3; draw z3 .. z4 .. z5 .. z6 .. z3;                                          % bowl
endchar;

beginpunkchar(oct "011", 14, 1, 1);                                              % Ψ
z1 = pp(.5w, h); z2 = pp(.5w, 0); pd z1; pd z2; draw z1 -- z2;                   % stem
z3 = pp(u, .8h); z4 = pp(.5w, .2h); z5 = pp(w - u, .8h);
pd z3; pd z5; draw z3{.4[z1, z2] - z3} .. z4{right} .. {z5 - .4[z1, z2]}z5;      % stroke
endchar;

beginpunkchar(oct "012", 13, 1, 2);                                              % Ω
z1 = pp(u, 0); z2 = pp(1/3w, 0); z3 = pp(u, 2/3h); z4 = pp(.5w, h);
z5 = pp(w - u, 2/3h); z6 = pp(2/3w, 0); z7 = pp(w - u, 0);
pd z1; pd z7; draw z1 -- z2{up} .. z3 .. z4 .. z5 .. {down}z6 -- z7;             % bowl and arms
endchar;
```

The next program file, PUNKD.MF, defines the ten punk digits. I ran out of time while typing this, so the comments at the end are somewhat uninspired:

```
beginpunkchar("0", 9, .5, 1);
z1 = pp(.5w, h); z2 = pp(u, .55h); z3 = pp(.5w, 0); z4 = pp(w - u, .55h);
pd z1; draw z1{curl 2} .. z2 .. z3 .. z4 .. z1;                                  % bowl
endchar;

beginpunkchar("1", 9, .3, 1);
z1 = pp(2u, .7h); z2 = pp(.6w, h); z3 = pp(.6w, 0);
pd z1; pd z3; draw z1 -- z2 -- z3;                                               % serif and stem
endchar;

beginpunkchar("2", 9, 1, 1);
z1 = pp(2u, .7h); z2 = pp(.5w, h); z3 = pp(w - u, .6h); z4 = pp(u, 0); z5 = pp(w - 2u, 0);
pd z1; pd z5; draw z1 .. z2 .. z3 .. z4 -- z5;                                   % stroke
endchar;

beginpunkchar("3", 9, .5, .5);
z1 = pp(2u, .7h); z2 = pp(.5w, h); z3 = pp(w - u, .5[y2, y4]);
z4 = pp(.5w - u, .55h); z5 = pp(w - u, .5[y4, y6]); z6 = pp(.5w, 0); z7 = pp(1.5u, .2h);
pd z1; pd z7; draw z1 .. z2 .. z3 .. z4 & z4 .. z5 .. z6 .. z7;                  % arcs
endchar;

beginpunkchar("4", 9, 1, 1);
z1 = pp(w - u, .3h); z2 = pp(u, .3h); z3 = pp(2/3w, h); z4 = pp(2/3w, 0);
pd z1; pd z4; draw z1 -- z2 -- z3 -- z4;                                         % stem and diagonals
endchar;

beginpunkchar("5", 9, .5, .5);
z1 = pp(w - 2u, h); z2 = pp(2u, h); z3 = pp(u, .7h); z4 = pp(w - u, .5[y3, y5]);
z5 = pp(.5w, 0); z6 = pp(u, .2h);
pd z1; pd z6; draw z1 -- z2 -- z3 .. z4 .. z5 .. z6;                             % stroke
endchar;

beginpunkchar("6", 9, 1, 1);
z1 = pp(2/3w, h); z2 = pp(u, .3h); z3 = pp(.5w, 0); z4 = pp(w - u, .3h); z5 = pp(.6w, .6h);
z6 = pp.z 2; pd z1; pd z6; draw z1 .. z2 .. z3 .. z4 .. z5 -- z6;                % stroke
endchar;
```

```
beginpunkchar("7", 9, .5, 1);
z1 = pp(2u, h); z2 = pp(w - .5u, h); z3 = pp(.4w, 0);
pd z1; pd z3; draw z1 -- z2 & z2 .. z3{down};                          % stroke
endchar;
beginpunkchar("8", 9, .5, .5);
z1 = pp(.5w, h); z2 = pp(u, .5[y1, y3]); z3 = pp(.5w, .6h); z4 = pp(w - u, .5[y3, y5]);
z5 = pp(.5w, 0); z6 = pp(u, .5[y5, y3]); z7 = pp(w - u, .5[y1, y3]);
pd z1; draw z1{curl 8} .. z2 .. z3 .. z4 .. z5 .. z6 .. z3 .. z7 .. z1;   % stroke
endchar;
beginpunkchar("9", 9, 1, 1);
z1 = pp(1/3w, 0); z2 = pp(w - u, .7h); z3 = pp(.5w, h); z4 = pp(u, .7h); z5 = pp(.5w, .4h);
pd z1; pd z5; draw z1 .. z2 .. z3 .. z4 .. z5;                         % stroke
endchar;
```

The program file PUNKP.MF defines "punk punctuation." This was one of the most difficult to write—although most of the characters are very simple—because there are so DARN many punctuation marks.

```
beginpunkchar(".", 5, 1, 2);
pd pp(.5w, 0);                                                         % dot
endchar;
beginpunkchar(",", 5, .5, .5);
z1 = pp(.5w, 0); z2 = pp(w - u, -.1h); z3 = pp(.5w, -.3h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                     % stroke
endchar;
beginpunkchar(":", 5, 1, .5);
pd pp(.5w, 0); pd pp(.5w, .4h);                                        % dots
endchar;
beginpunkchar(";", 5, .5, .5);
z1 = pp(.5w, 0); z2 = pp(w - u, -.1h); z3 = pp(.5w, -.3h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                     % stroke
pd pp(.5w, .4h);                                                       % dot
endchar;
beginpunkchar("!", 5, .5, .5);
pd pp(.5w, 0);                                                         % dot
z1 = pp(.5w, 1.05h); z2 = pp(.5w, .3h); pd z1; pd z2; draw z1 -- z2;   % stem
endchar;
ligtable "!": "‘" =: oct "016";
beginpunkchar(oct "016", 5, .5, .5);                     % Spanish inverted !
pd pp(.5w, .9h);                                                       % dot
z1 = pp(.5w, -.1h); z2 = pp(.5w, .6h); pd z1; pd z2; draw z1 -- z2;    % stem
endchar;
beginpunkchar("?", 9, 1, .5);
z1 = pp(1.5u, .8h); z2 = pp(.5w, h); z3 = pp(w - u, .8h); z4 = pp(.5w, .3h);
pd z1; pd z4; draw z1 .. z2 .. z3 .. {down}z4;                  % arc and stem
pd pp(.5w, 0);                                                         % dot
endchar;
ligtable "?": "‘" =: oct "017";
beginpunkchar(oct "017", 9, 1, .5);                      % Spanish inverted ?
z1 = pp(1.5u, .1h); z2 = pp(.5w, -.1h); z3 = pp(w - u, .1h); z4 = pp(.5w, .6h);
pd z1; pd z4; draw z1 .. z2 .. z3 .. {up}z4;                    % arc and stem
pd pp(.5w, .9h);                                                       % dot
endchar;
```

```
beginpunkchar("&", 14, .5, .5);
z1 = pp(w - 2u, h); z2 = pp(u, h); z3 = pp(3u, 0); z5 = pp(w - u, .6h); z6 = pp(w - 2u, 0);
pd z1; pd z5; draw z1 -- z2 -- z3 -- z5;                                  % arms and stem
draw z1 -- .5[z2, z3]; pd z6; draw z6 -- .6[z3, z5];                          % diagonals
endchar;

beginpunkchar("$", 12, .5, .5);
z1 = pp(w - 1.5u, .7h); z2 = pp(.5w, h); z3 = pp(u, .7h); z4 = .5[z3, z5];
z5 = pp(w - u, .3h); z6 = pp(.5w, 0); z7 = pp(u, .3h);
pd z1; pd z7; draw z1 .. z2 .. z3 .. z4 .. z5 .. z6 .. z7;                        % stroke
z8 = z2 + (0, .1h); pd z8; draw z8 -- z6;                                           % stem
endchar;

beginpunkchar("%", 18, .5, .5);
z1 = pp(3.5u, 1.1h); z2 = pp(u, .8h); z3 = pp(3.5u, .5h); z4 = pp(6u, .8h);
z5 = pp(w - 3.5u, .5h); z6 = pp(w - 6u, .2h); z7 = pp(w - 3.5u, -.1h); z8 = pp(w - u, .2h);
pd z1; draw z1 .. z2 .. z3 .. z4 .. z1;                                       % upper bowl
pd z5; draw z5 .. z6 .. z7 .. z8 .. z5;                                       % lower bowl
z9 = pp(w - 3u, 1.1h); z0 = pp(3u, -.1h); pd z0; draw z9 -- z0;                   % diagonal
draw z1{z5 - z1} .. z9;                                                             % link
endchar;

beginpunkchar("@", 18, 1, .5);
z1 = pp(2u, 0); z2 = pp(1/3w, .7h); z3 = pp(w - 6u, 0);
z4 = pp(w, .3h); z5 = pp(1/3w, h); z6 = pp(u, .5h); z7 = .7[z2, z3];
pd z1; pd z7; draw z1 -- z2 -- z3{right} .. z4 .. z5 .. z6 .. z7;     % diagonals and stroke
endchar;

beginpunkchar("-", 7, .5, .5);
z1 = pp(u, .4h); z2 = pp(w - u, .5h); pd z1; pd z2; draw z1 -- z2;                  % bar
endchar;
ligtable "-": "-" =: oct "173";

beginpunkchar(oct "173", 9, .5, .5);                                                  % –
z1 = pp(0, .5h); z2 = pp(w, .4h); pd z1; pd z2; draw z1 -- z2;                       % bar
endchar;
ligtable oct "173": "-" =: oct "174";

beginpunkchar(oct "174", 18, .5, .5);                                                 % —
z1 = pp(0, .5h); z2 = pp(w, .4h); pd z1; pd z2; draw z1 -- z2;                       % bar
endchar;

beginpunkchar("+", 9, .5, 1);
z1 = pp(0, .5h); z2 = pp(w, .5h); pd z1; pd z2; draw z1 -- z2;                       % bar
z3 = pp(.5w, .1h); z4 = pp(.5w, .9h); pd z3; pd z4; draw z3 -- z4;                  % stem
endchar;

beginpunkchar("*", 13, .5, 1);
z0 = pp(.5w, 1.1h); z1 = pp(u, .9h); z2 = pp(2u, .3h); z3 = pp(w - u, .3h); z4 = pp(w - u, .9h);
pd z0; draw z0 -- z2 .. 1/3[.5[z2, z4], z0] .. z4 -- z1 -- z3 -- z0;                 % star
endchar;

beginpunkchar("'", 5, .3, .5);
z1 = pp(1.5u, h); z2 = pp(w - u, .85h); z3 = pp(u, 2/3h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                                % stroke
endchar;
ligtable "'": "'" =: oct "042";
```

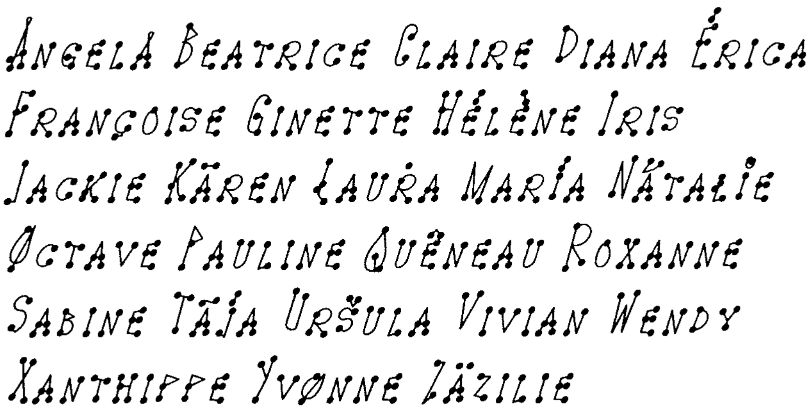
```
beginpunkchar(oct "042", 9, .3, .5);                                             % ”
z1 = pp(.5w − .5u, h);  z2 = pp(u, .6h);  z3 = pp(w − u, .95h);
pd z1;  pd z3;  draw z1 -- z2 -- z3;                                             % stroke
endchar;

beginpunkchar("‘", 5, .3, .5);
z1 = pp(w − 1.5u, h);  z2 = pp(u, .85h);  z3 = pp(w − u, 2/3h);
pd z1;  pd z3;  draw z1 -- z2 -- z3;                                             % stroke
endchar;
ligtable "‘": "‘" =: oct "134";

beginpunkchar(oct "134", 9, .3, .5);                                             % “
z1 = pp(.5w + .5u, h);  z2 = pp(w − u, .6h);  z3 = pp(u, .95h);
pd z1;  pd z3;  draw z1 -- z2 -- z3;                                             % stroke
endchar;

beginpunkchar(oct "015", 9, .3, .5);                                             % '
z1 = pp(.5w, h);  z2 = pp(.5w, .6h);  pd z1;  pd z2;  draw z1 -- z2;             % stem
endchar;

beginpunkchar("(", 7, .5, .5);
z1 = pp(w − u, h);  z2 = pp(u, .5h);  z3 = pp(w − u, 0);  pd z1;  pd z3;  draw z1 .. z2 .. z3;   % stroke
endchar;

beginpunkchar(")", 7, .5, .5);
z1 = pp(u, h);  z2 = pp(w − u, .5h);  z3 = pp(u, 0);  pd z1;  pd z3;  draw z1 .. z2 .. z3;       % stroke
endchar;

beginpunkchar("[", 8, .5, .5);
z1 = pp(w − u, h);  z2 = pp(.5w, h);  z3 = pp(.5w, 0);  z4 = pp(w − u, 0);
pd z1;  pd z4;  draw z1 -- z2 -- z3 -- z4;                                       % bars and stem
endchar;

beginpunkchar("]", 8, .5, .5);
z1 = pp(u, h);  z2 = pp(.5w, h);  z3 = pp(.5w, 0);  z4 = pp(u, 0);
pd z1;  pd z4;  draw z1 -- z2 -- z3 -- z4;                                       % bars and stem
endchar;

beginpunkchar("<", 9, .5, .5);
z1 = pp(w − u, .9h);  z2 = pp(u, .5h);  z3 = pp(w − u, .1h);
pd z1;  pd z3;  draw z1 -- z2 -- z3;                                             % diagonals
endchar;

beginpunkchar(">", 9, .5, .5);
z1 = pp(u, .9h);  z2 = pp(w − u, .5h);  z3 = pp(u, .1h);
pd z1;  pd z3;  draw z1 -- z2 -- z3;                                             % diagonals
endchar;

beginpunkchar("=", 9, .5, .5);
z5 = pp(u, 2/3h);  z6 = pp(w − u, 2/3h);  pd z5;  pd z6;  draw z5 -- z6;         % upper bar
z7 = pp(u, 1/3h);  z8 = pp(w − u, 1/3h);  pd z7;  pd z8;  draw z7 -- z8;         % lower bar
endchar;

beginpunkchar("#", 15, .5, .5);
z1 = pp(.5w, h);  z2 = pp(3u, 0);  z3 = pp(w − 3u, h);  z4 = pp(.5w, 0);
pd z2;  pd z3;  draw z3 -- z1 -- z2;  draw z3 -- z4 -- z2;                       % diagonals (linked)
z5 = pp(u, 2/3h);  z6 = pp(w − u, 2/3h);  pd z5;  pd z6;  draw z5 -- z6;         % upper bar
z7 = pp(u, 1/3h);  z8 = pp(w − u, 1/3h);  pd z7;  pd z8;  draw z7 -- z8;         % lower bar
endchar;
```

```
beginpunkchar("/", 9, 1, 1);
z1 = pp(1.5u, -.05h); z2 = pp(w - 1.5u, 1.05h); pd z1; pd z2; draw z1 -- z2;     % diagonal
endchar;
beginpunkchar(oct "013", 12, .5, .5);                                              % ↑
z1 = pp(u, .7h); z2 = pp(.5w, h); z3 = pp(w - u, .7h); z4 = pp(.5w, 0);
pd z1; pd z3; pd z4; draw z1 -- z2 -- z4; draw z3 -- z2;                 % stem and diagonals
endchar;
beginpunkchar(oct "014", 12, .5, .5);                                              % ↓
z1 = pp(u, .3h); z2 = pp(.5w, 0); z3 = pp(w - u, .3h); z4 = pp(.5w, h);
pd z1; pd z3; pd z4; draw z1 -- z2 -- z4; draw z3 -- z2;                 % stem and diagonals
endchar;
```

The final program file, PUNKA.MF, defines accents in a form that TEX likes. The TEX input

```
\def\AA{\accent'27A}
{\AA}ngel\aa\ Beatrice Claire Diana \'Erica Fran\c{c}oise
Ginette H\'el\`ene Iris Jackie K\=aren {\L}au\.ra Mar{\'\i}a
N\H{a}ta{\l}{\u\i}e {\O}ctave Pauline Qu\^eneau Roxanne Sabine
T\~a{\'\j}a Ur\v{s}ula Vivian Wendy Xanthippe Yv{\o}nne Z\"azilie
```

causes accents to be positioned as follows, in the font PUNKSL20:


ÅNGELÅ BEATRICE CLAIRE DIANA ÉRICA
FRANÇOISE GINETTE HÉLÈNE IRIS
JACKIE KĀREN ŁAURȦ MARÍA NŐTAŁĬE
ØCTAVE PAULINE QUÊNEAU ROXANNE
SABINE TÃȷ́A URŠULA VIVIAN WENDY
XANTHIPPE YVØNNE ZÄZILIE

(Notice that the macro \AA needs to be redefined, but the other accents of plain TEX work without change.)

Here is the way accents are drawn:

```
beginpunkchar(oct "022", 9, 1, 1);                                                 % `
z1 = pp(2.5u, h); z2 = pp(.6w, .8h); pd z1; pd z2; draw z1 -- z2;                 % diagonal
endchar;
beginpunkchar(oct "023", 9, 1, 1);                                                 % ´
z1 = pp(w - 2.5u, h); z2 = pp(.4w, .8h); pd z1; pd z2; draw z1 -- z2;             % diagonal
endchar;
beginpunkchar(oct "136", 13, 1, 1);                                                % ˆ
z1 = pp(2.5u, .8h); z2 = pp(.5w, h); z3 = (w - 2.5u, .8h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                                 % diagonals
endchar;
beginpunkchar(oct "024", 13, 1, 1);                                                % ˘
z1 = pp(2.5u, .9h); z2 = pp(.5w, .7h); z3 = pp(w - 2.5u, .9h);
pd z1; pd z3; draw z1 -- z2 -- z3;                                                 % diagonals
endchar;
```

```
beginpunkchar(oct "025", 11, 1, 1);                                                % ˘
z1 = pp(2u, h); z2 = pp(.5w, .75h); z3 = pp(w - 2u, h);
pd z1; pd z3; draw flex(z1, z2, z3);                                               % stroke
endchar;
beginpunkchar(oct "026", 12, 1, 1);                                                % ¯
z1 = pp(u, .8h); z2 = pp(w - u, .8h); pd z1; pd z2; draw z1 -- z2;                 % bar
endchar;
beginpunkchar(oct "137", 5, 1, 1);                                                 % ˙
pd pp(.5w, .9h);                                                                   % dot
endchar;
beginpunkchar(oct "177", 13, 1, 1);                                                % ¨
pd pp(1/5w, .9h); pd pp(4/5w, .9h);                                                % dots
endchar;
beginpunkchar(oct "176", 13, 1, 1);                                                % ˜
z1 = pp(u, .75h); z2 = pp(w - u, .9h); pd z1; pd z2; draw z1{up} .. {up}z2;        % stroke
endchar;
beginpunkchar(oct "175", 13, 1, 1);                                                % ˝
z1 = pp(4u, h); z2 = pp(2.5u, .7h); z3 = pp(w - 2u, h); z4 = pp(w - 3.5u, .7h);
pd z1; pd z3; draw z1 -- z2 -- z4 -- z3;                                           % diagonals (linked)
endchar;
beginpunkchar(oct "027", 13, 0, 0);                      % Scandinavian loop, for Å and å
z0 = (.5w, .66h);                                                   % point z2 of lowercase A
z1 = (.5w, .9h); draw z0{z0 - (1.5u, 0)} .. z1 .. {(w - 1.5u, 0) - z0}z0;          % loop
endchar;
beginpunkchar(oct "030", 13, .5, .5);                                              % Cedilla, for ç
z1 = (.6w, 0); z2 = pp(.6w, -.1h); z3 = pp(2.5u, -.1h);
pd z3; draw z1 -- z2 -- z3;                                                        % stroke
endchar;
beginpunkchar(oct "040", 11, .5, .5);                              % Polish cross, for Ł and ł
z1 = pp(0, .25h); z2 = pp(4u, .4h); pd z1; pd z2; draw z1 -- z2;                   % diagonal
endchar;
ligtable oct "040": "l" kern -charwd, "L" kern -charwd;
```

Bibliography

[1] George Ade, *Artie: A Story of the Streets and Town*, 1896, Chapter 19.

[2] Dashiell Hammett, *The Maltese Falcon*, 1930, Chapter 18.

[3] Derek Jewell, music review in the *Sunday Times*, 28 November 1976, page 37.

[4] Donald E. Knuth, *The METAFONTbook*, Addison-Wesley, 1986, page v.

[5] Pablo Picasso, from his sketchbooks. (This drawing was later used as an illustration in Vollard's de luxe edition of *Le Chef-d'Œuvre Inconnu* by Honoré de Balzac, 1931.)

[6] William Shakespeare, *The Merry Wives of Windsor*, Act 2, Scene 2, lines 135–137. (The *First Folio* has the spelling 'Puncke'.)

Output Devices

TEX Output Devices

Don Hosek

The device tables on the following pages list all the TEX device drivers currently known to TUG. Some of the drivers indicated in the tables are considered proprietary. Most are not on the standard distribution tapes; those drivers which are on the distribution tapes are indicated in the listing of sources below. To obtain information regarding an interface, if it is supposed to be included in a standard distribution, first try the appropriate site coordinator or distributor; otherwise request information directly from the sites listed.

The codes used in the charts are interpreted below, with a person's name given for a site when that information could be obtained and verified. If a contact's name appears in the current TUG membership list, only a phone number or network address is given. If the contact is not a current TUG member, the full address and its source are shown. When information on the drivers is available, it is included below.

Screen previewers for multi-user computers are listed in the section entitled "Screen Previewers". If a source has been listed previously under "Sources", then a reference is made to that section for names of contacts.

Corrections, updates, and new information for the list are welcome; send them to Don Hosek, Bitnet Dhosek@Hmcvax (postal address, page 115).

Sources

ACC Advanced Computer Communications, Diane Cast, 720 Santa Barbara Street, Santa Barbara, CA 93101, 805-963-9431 (DECUS, May '85)

Adelaide Adelaide University, Australia

The programs listed under Adelaide have been submitted to the standard distributions for the appropriate computers. The PostScript driver permits inclusion of PostScript files in a TEX file. The driver is described in *TUGboat, Vol. 8, No. 1.*

AMS American Mathematical Society, Barbara Beeton, 401-272-9500 Arpanet: BNB@Seed.AMS.com

Arbor ArborText, Inc., Bruce Baker, 313-996-3566, Arpanet: Bwb@Arbortext.Com

ArborText's software is proprietary and ranges in price from $150 to $3000. The drivers for PostScript printers, the HP LaserJet Plus, the QMS Lasergrafix, and Imagen printers are part of their DVILASER series. The drivers all support graphics and include other special features such as use of resident fonts or landscape printing when supported by the individual printers.

Printing on the Autologic APS-5 and μ-5 phototypesetters with DVIAPS includes support of Autologic standard library fonts and logo processing.

Bochum Ruhr Universität Bochum, Norbert Schwarz, 49 234 700-4014

Caltech California Institute of Technology, Chuck Lane, Bitnet: CEL@CITHEX

Canon Canon Tokyo, Masaaki Nagashima, (03)758-2111

Carleton Carleton University, Neil Holtz, 613-231-7145

CMU Carnegie-Mellon University, Howard Gayle, 412-578-3042

Columb. Columbia University, Frank da Cruz, 212-280-5126

COS COS Information, Gilbert Gingras, 514-738-2191

DEC Digital Equipment Corporation, John Sauter, 603-881-2301

The LN03 driver is on the VAX/VMS distribution tape.

ENS Ecole Normale Superieure, Chantal Durand,
Centre de Calcul, Ecole Normale Superieure,
45 rue d'Ulm, 75005 Paris, France

GA Tech GA Technologies

GMD1 Gesellschaft für Mathematik und Datenverarbeitung, Federal Republic of Germany, Ferdinand Hommes, Bitnet: Grztex@Dbngmd21, 0228-303221

GMD2 Gesellschaft für Mathematik und Datenverarbeitung, Federal Republic of Germany, Dr. Wolfgang Appelt, uucp: seismo!unido!gmdzi!zi.gmd.dbp.de!appelt

Heidelb'g University of Heidelberg, Federal Republic of Germany, Joachim Lammarsch, Bitnet: Rz92@Dhdurdz1

HMC Harvey Mudd College, Don Hosek, Bitnet: Dhosek@Ymir

HP Hewlett-Packard, Stuart Beatty, 303-226-3800

INFN INFN/CNAF, Bologna, Italy, Maria Luisa Luvisetto, 51-498286, Bitnet: MiLtex@Iboinfn

The CNAF device drivers are on the VAX/VMS distribution tape.

Interg'ph Intergraph, Mike Cunningham, 205-772-2000

JDJW JDJ Wordware, John D. Johnson, 415-965-3245, Arpanet: M.John@Sierra.Stanford.Edu

Kettler Kettler EDV Consulting, P. O. Box 1345,
D-8172 Lenggries, Federal Republic Germany,
+49 8042 8081

The LaserJet driver supports graphics inclusion in device dependent format. PK font files are used. This program is proprietary. Contact Kettler for further information.

LaserPrint LaserPrint, P. O. Box 35, D-6101
Fränkisch Crumbach, Federal Republic Germany,
+49 6164 4044

The driver supports graphics inclusion in device dependent format. PK font files are used. This program is proprietary. Contact LaserPrint for further information.

LLL Lawrence Livermore Laboratory

LSU Louisiana State University, Neal Stoltzfus,
504-388-1570

Milan1 Università Degli Studi Milan, Italy,
Dario Lucarella, 02/23.62.441

Milan2 Università Degli Studi Milan, Italy,
Giovanni Canzii, 02/23.52.93

MIT Massachusetts Institute of Technology,
Chris Lindblad, MIT AI Laboratory, 617-253-8828

The drivers for Symbolics Lisp machines use the Symbolics Generic Hardcopy interface as a back end, so it should work on any printer that has a driver written for it. The printers listed in the table indicate drivers the program has been tested on.

The UNIX drivers for PostScript and QMS printers both support landscape printing and graphics inclusion via specials.

MPAE Max-Planck-Institut für Aeronomie,
H. Kopka, (49) 556-41451, Bitnet: `Mio40L@D606wd01`

MR Math Reviews, Dan Latterner, 313-996-5266

NLS Northlake Software, David Kellerman,
503-228-3383

The VAX/VMS Imagen driver supports graphics.

OCLC OCLC, Thom Hickey, 6565 Frantz Road,
Dublin, OH 43017, 616-764-6075

OSU1 Ohio State University, John M. Crawford,
614-292-1741, Bitnet: `Ts0135@Ohstvma`,
Internet: `Crawford-j@Ohio-state.Edu`

OSU2 Ohio State University, Ms. Marty Marlatt,
Department of Computer and Information Science,
2036 Neil Avenue, Columbus, OH 43210

The drivers are distributed on either ANSI or TOPS-20 DUMPER tapes, with hardcopy documentation. There is a $125 service charge (payable to Ohio State University) to cover postage, handling, photocopying, etc.

Pers Personal TeX, Inc., Lance Carnes,
415-388-8853

Graphics output is supported on Imagen, PostScript, and QMS printers.

Philips Philips Kommunikations Industrie AG,
TEKADE Fernmeldeanlagen, Attn. Dr. J. Lenzer,
Thurn-und-Taxis-Str., D-8500 Nürnberg,
Federal Republic Germany, +49 911 5262019

PPC Princeton Plasma Physics Lab, Charles
Karney, Arpanet: `Karney%PPC.MFENET@NMFECC.ARPA`

Versatec output from TeXspool is produced via the NETPLOT program. TeXspool also produces output for the FR80 camera. Color and graphics primitives are supported through specials.

Procyon Procyon Informatics, Dublin, Ireland,
John Roden, 353-1-791323

RTI Research Triangle Institute, Randy Buckland,
Arpanet: `rcb@rti.rti.org`

The program is available in the `comp.sources.misc` archives on Arpanet and Usenet.

Saar Universität des Saarlandes, Saarbrücken,
Federal Republic of Germany, Prof. Dr. Reinhard
Wilhelm, uucp: `wilhelm@sbsvax.UUCP`

SARA Stichting Acad Rechenzentrum Amsterdam,
Han Noot, Stichting Math Centrum,
Tweede Boerhaavestraat 49, 1091 AL Amsterdam
(see *TUGboat, Vol. 5, No. 1*)

Scan Scan Laser, England, John Escott,
+1 638 0536

Sci Ap Science Applications, San Diego, CA,
619-458-2616

SEP Systemhaus für Elektronisches Publizieren,
Robert Schöninger, Arndtstrasse 12, 5000 Köln,
Federal Republic of Germany

DVIP400 uses PXL files. Landscape printing is supported in all versions and graphics inclusion in all but the IBM PC version. Source is available on request. Cost varies from 300–1848DM.

Stanford Stanford University

The Imagen driver from Stanford is present on most distributions as the file `DVIIMP.WEB`. It provides limited graphics ability.

Sun Sun, Inc.

Sydney University of Sydney, Alec Dunn,
(02) 692 2014, ACSnet: `alecd@facet.ee.su.oz`

Talaris Talaris, Rick Brown, 619-587-0787

All of the Talaris drivers support graphics.

T A&M1 Texas A&M, Bart Childs, 409-845-5470,
CSnet: `Childs@TAMU`

Graphics is supported on the Data General drivers for the Printronix, Toshiba, and Versatec on the Data General MV. On the TI PC, graphics is supported on the Printronix and Texas Instruments 855 printers. There are also previewers available for both the Data General and the TI.

T A&M2 Texas A&M, Ken Marsh, 409-845-4940,
Bitnet: `KMarsh@TAMNIL`

T A&M3 Texas A&M, Norman Naugle,
409-845-3104

The QMS driver supports inclusion of QUIC graphics commands via specials as well as landscape printing.

T A&M4 Texas A&M, Thomas Reid, 409-845-8459,
Bitnet: `X066TR@TAMVM1`

The TeXrox package includes a GF/PK/PXL to Xerox font converter (PXLrox2), and utility to build TFM files from licensed Xerox fonts (Xetrix). The programs are all written in C. Fonts not present on the Xerox printers can be printed as bitmaps on printers with the graphics handling option (GHO).

At present the TeXrox package is being distributed on a twelve-month trial basis; the trial is free for U.S. educational and government institutions, $100 for foreign or commercial institutions. Licensing agreements will be available when the trial offer expires.

THD Technische Hochschule Darmstadt,
Klaus Guntermann, Bitnet: `XITIKGUN@DDATHD21`

The program uses PK fonts. The Philips Elpho driver is not public domain. Contact Klaus Guntermann for information on obtaining the program.

Tools Tools GmbH Bonn, Edgar Fuß,
Kessenicher Straße 108, D-5300 Bonn 1,
Federal Republic of Germany

The Tools implementation of TeX and the drivers listed are described in *TUGboat, Vol. 8, No. 1.*

TRC Finl'd Technical Research Centre of Finland,
Tor Lillqvist, +358 0 4566132, Bitnet: `tml@fingate`

UBC University of British Columbia, Afton Cayford,
604-228-3045

UCB University of California, Berkeley,
Michael Harrison, Arpanet: `vortex@berkeley.arpa`

UCIrv1 University of California, Irvine,
David Benjamin

UCIrv2 University of California, Irvine,
Tim Morgan, Arpanet: `Morgan@UCI.ARPA`

U Del University of Delaware, Daniel Grim,
302-451-1990, Arpanet: `grim@huey.udel.edu`

The distribution includes a program to convert font files generated by METAFONT to Xerox font format.

U Ill University of Illinois, Dirk Grunwald,
Arpanet: `Grunwald@M.Cs.Uiuc.Edu`

The previewers are available via anonymous FTP in the directory `pub/iptex.tar.Z` on `a.cs.uiuc.edu`.

U Köln Univ of Köln, Federal Republic of
Germany, Jochen Roderburg, 0221-/478-5372,
Bitnet: `A0045@DkOrrzk0`

U Mass University of Massachusetts, Amherst,
Gary Wallace, 413-545-4296

U MD University of Maryland, Chris Torek,
301-454-7690, Arpanet: `chris@mimsy.umd.edu`

The UNIX Imagen driver is on the UNIX distribution tape. The drivers may be obtained via anonymous FTP from `a.cs.uiuc.edu` in the directory `pub/iptex.tar.Z` or from `mimsy.umd.edu` in the directory `tex`.

U Mich University of Michigan, Kari Gluski,
313-763-6069

UNI.C Aarhus University, Regional Computer
Center, Denmark

U Shef University of Sheffield, England,
Ewart North, (0742)-78555, ext. 4307

Utah University of Utah, Nelson H. F. Beebe,
801-581-5254, Arpanet: `Beebe@Science.Utah.edu`

All of the Beebe drivers are distributed together. They are available on IBM PC-DOS floppy disks (about 6), or 1600bpi 9-track tape in TOPS-10/20 BACKUP/DUMPER format, VAX/VMS BACKUP format, Unix tar format, and ANSI D-format. Send tape or disks for a copy. The programs are available for anonymous FTP from `SCIENCE.UTAH.EDU` on the internet; information is in the file `PS:<ANONYMOUS>OOREADME.TXT`. A VAX/VMS binary distribution is available for anonymous FTP (password guest) from `CTRSCI.UTAH.EDU`. `OOREADME.TXT` in the login directory gives details. On JANET, the programs may be obtained from the directory `aston.kirk::[public.texdvi210]`. The drivers are available from Listserv on EARN to European Bitnet users. Sending the command `GET DRIVER FILELIST` (in an interactive message, or as the first line of a mail message) to `LISTSERV@DHDURZ1`. Files are obtained with the command `GET` *filename filetype.* Graphics is supported only in the DVIALW (PostScript) driver.

U Wash1 University of Washington,
Pierre MacKay, 206-543-6259,
Arpanet: `MacKay@June.CS.Washington.edu`

The programs listed under U Wash1 are all on the standard UNIX distribution tape.

U Wash2 University of Washington, Jim Fox,
206-543-4320, Bitnet: `fox7632@uwacdc`

The QMS driver for the CDC Cyber was written under NOS 2.2 and supports graphics.

Vander Vanderbilt University, H. Denson Burnum,
615-322-2357

Wash St Washington State University, Dean
Guenther, 509-335-0411, Bitnet: `Guenther@Wsuvm1`

Wash U Washington University, Stanley Sawyer,
314-889-6703

The IBM PC LN03 driver is a modified version of Flavio Rose's `DVI2LN3`. Graphics support is provided through inclusion of LN03 plotfiles and line drawing specials. All three PXL formats on the PC are supported. The program is available free of charge with the receipt of a blank disk and return mailer.

W'mann Weizmann Institute, Rehovot,
Israel, Malka Cymbalista, 08-482443,
Bitnet: `Vumalki@Weizmann`

Xerox Xerox, Margaret Nelligan, Xerox
Printing Systems Division, 880 Apollo Street,
El Segundo, CA 90245, 213-333-6058

Yale Yale University, Jerry Leichter,
Arpanet: `Leichter-jerry@Cs.Yale.Edu`,
Bitnet: `Leichter@Yalevms`

DVIDIS is available for anonymous FTP from `Venus.Ycc.Yale.Edu`. Log in as anonymous and do a `CD [.DVIDIS]`. That directory contains the three required files needed to run the previewer. The image *must* be transferred using BINARY mode.

Screen Previewers — Multi User Systems

▪ **Data General MV**

T A&M1

▪ **DEC-20**

OSU2 ASCII Output

Utah BBN Bitgraph terminal

▪ **HP9000/500**

Utah BBN Bitgraph terminal

▪ **IBM MVS**

GMD GDDM supported devices: IBM 3179, 3192, 3193, and 3279

Milan1 Tektronix 4014

▪ **IBM VM/CMS**

HMC Terminals connected through 7171 Protocol converters: Tektronix compatible, VT-640 compatible, GDDM driven IBM 3179 and 3279 terminals,
GDDM driven Tektronix 816

DVIview may be obtained by sending $30 (to defray duplication costs), a blank tape, and a return mailer to Don Hosek. The program is still in the developmental stages, and enhancements will be made in the future. The program uses PK files.

Wash St GDDM driven IBM 3179 and 3279 terminals

Uses PXL files at 120dpi. Allows viewing of the page in eight parts normal size or three parts compressed.

W'mann IBM 3279, 3179-G

Previewing is provided by DVI82, the Weizmann driver for the Versatec plotter. The program uses PXL files.

▪ **UNIX**

Talaris Talaris 7800

Utah BBN Bitgraph

U Wash1 DMD5620

Uses GF, PK, or PXL files at 118dpi. `tpic` output is supported. The program consists of two parts: a program running on the host computer and another that is downloaded to the terminal.

▪ **VAX VMS**

Adelaide AED 512, ANSI-compatible, DEC ReGIS, DEC VT100, DEC VT220, Tektronix 4014,
Visual 500, 550

Uses PK or PXL files.

INFN DEC ReGIS

Uses PXL files.

Talaris Talaris 7800

Utah BBN Bitgraph

Screen Previewers — Microcomputers and Workstations

▪ **Apollo**

Arbor

Uses GF, PK, and PXL files. Preview is available for $500.

U Ill X-11 Windows System

▪ **Atari ST**

Kettler

Tools

▪ **Cadmus 9200**

U Köln

▪ **IBM PC**

Arbor, Pers EGA, MCGA, UGA, Hercules, Olivetti, Tecmar, Genius full page, ETAP Neftis, Toshiba 3100, AT&T 6300

Uses GF, PK, and PXL files as well as tuned PostScript fonts (the base set available with PostScript printers). Preview of integrated bit map graphics is supported. Preview is available for $175.

T A&M3 EGA, CGA, Hercules

The `cdvi` program is available for $175.

▪ **IBM PC/RT**

U Ill X-11 Windows

▪ **Integrated Solutions**

UCIrv1

Utah BBN Bitgraph

▪ **SUN**

Arbor

Uses GF, PK, and PXL files. Preview is available for $500.

UCB

UCIrv2

U Ill X-11 Windows, Sunview Window System

Uses GF, PK, and PXL files.

▪ **Vaxstation/Unix**

U Ill X-11 Windows

Uses GF, PK, and PXL files.

▪ **Vaxstation/VMS**

Arbor GPX(UIS)

Uses GF, PK, and PXL files.
Preview is available for $500.

INFN GPX(UIS)

Uses PXL files.

Philips GPX(UIS)

RTI GPX(UIS)

Uses PK files at 78, 94 and 112dpi. Written in ADA. Source is included.

Yale GPX(UIS)

Uses PK files at 300dpi.

Low-Resolution Printers on Multi-User Systems — Laser Xerographic, Electro-Erosion Printers

	Amdahl (MTS)	CDC Cyber	Data General MV	DEC-10	DEC-20	HP9000 500	IBM MVS	IBM VM/CMS	IBM VM/UTS	Prime	Siemens BS2000	Sym-bolics Lisp	UNIX	VAX VMS
Agfa P400							SEP	SEP			Saar		Saar SEP	SEP
Canon					Utah	Utah							Canon Utah	Utah
DEC LN03					Utah	Utah							Utah	DEC NLS Procyon Utah
Golden Laser 100					Utah	Utah							Utah	Utah
HP LaserJet Plus					Utah	T A&M2 Utah				OSU1			Arbor Utah	Arbor Utah
IBM 38xx, 4250, Sherpa							GMD1 Heidelb'g	GMD1 Wash St						
Imagen	Arbor UBC		T A&M1	Stanford Vander	Columb. Utah	Utah	Arbor	Arbor W'mann				MIT	Arbor U Md Utah	Arbor NLS Utah
Philips Elpho														THD
PostScript printers					Utah	Arbor Utah		Arbor		OSU1		MIT	Arbor Carleton MIT Utah	Arbor Sydney Utah
QMS Lasergrafix	Arbor	U Wash2	T A&M1			T A&M2	Arbor GMD1	Arbor GMD1		OSU1 T A&M3	GMD1	MIT	Arbor MIT U Wash1	Arbor GA Tech T A&M3
Talaris							Talaris	Talaris					Talaris	Talaris
Xerox Dover					CMU								Stanford	
Xerox 2700II		Bochum			OSU2 Xerox			ENS					Xerox	
Xerox 9700	Arbor U Mich						Arbor T A&M4	Arbor T A&M4	T A&M4				U Del	ACC Arbor T A&M4

Low-Resolution Printers on Multi-User Systems — Impact and Electrostatic Printers

	CDC Cyber	Cray	Data General MV	DEC-10	DEC-20	HP9000 500	IBM MVS	IBM VM	Prime	UNIX	VAX VMS
Apple ImageWriter					Utah	Utah				Utah	LSU Utah
DEC LA75, LP100					OSU2 Utah	Utah				Utah	Utah
Epson					Utah	Utah				Utah	Utah
Facit 4542											INFN
Florida Data					MR						
MPI Sprinter					Utah	Utah				Utah	Utah
Okidata					Utah	Utah				Utah	Utah
Printronix			T A&M1		Utah	Utah				Utah	Utah
Toshiba			T A&M1		Utah	Utah				Utah	Procyon Utah
Varian											Sci Ap
Versatec	U Köln	PPC	T A&M1	GA Tech Vander	U Wash1		GMD1 U Milan2	W'mann	LLL	U Wash1	Caltech NLS

Low-Resolution Printers on Microcomputers and Workstations — Laser Xerographic, Electro-Erosion Printers

	Apollo	Atari ST	HP1000	HP3000	HP9000 200	IBM PC	Integrated Solutions	SUN
Agfa P400						SEP		
Canon		Utah				Utah	Utah	Utah
Cordata LP300						Pers		
DEC LN03		Utah				Utah Wash U	Utah	Utah
Golden Laser 100		Utah				Utah	Utah	Utah
HP 2680			JDJW	Pers				
HP 2688A			JDJW		HP			
HP LaserJet Plus	Arbor	Kettler Tools	TRC Finl'd		MPAE	Arbor Kettler Utah	Utah	Utah
Imagen	Arbor OCLC	Utah				Arbor Pers Utah	Utah	Arbor Sun U Md Utah
Kyocera		LaserPrint				LaserPrint		
Philips Elpho		Kettler						
PostScript printers	Arbor				Arbor	Arbor Pers Utah	Utah	Arbor MIT Utah
QMS Lasergrafix	Arbor Scan					Arbor Pers		Arbor MIT U Del
Talaris						Talaris		Talaris
Xerox 9700	COS Scan							T A&M4

Low-Resolution Printers on Microcomputers and Workstations — Impact and Electrostatic Printers

	Apollo	Atari ST	Cadmus 9200	HP1000	HP3000	IBM PC	Integrated Solutions	SUN
Apple ImageWriter		Utah				MR Utah	Utah	Utah
DEC LA75, LP100		Utah				Utah	Utah	Utah
Diablo					Pers			
Epson		Tools Utah		JDJW	U Shef	Milan1 Pers U Shef Utah	Utah	Utah
Fujitsu		Kettler	U Köln					
GE 3000	COS							
MPI Sprinter		Utah				Utah	Utah	Utah
NEC		Kettler						
Printronix		Utah				T A&M1 Utah	Utah	Utah
Star		Kettler						
Texas Instruments 855						T A&M1		
Toshiba		Utah				Pers Utah	Utah	Utah
Versatec								U Md

Typesetters

	Apollo	CDC Cyber	HP3000	IBM MVS	IBM PC	IBM VM/CMS	Siemens BS2000	Sperry 1100	SUN	UNIX	VAX VMS
Allied Linotype CRTronic											Procyon
Allied Linotype L100, L300P					Pers						
Allied Linotype L202					Pers						Procyon
Autologic APS-5, Micro-5	COS Scan				Arbor Pers				Arbor	Arbor	Arbor Interg'ph
Compugraphic 8400			U Shef		Arbor Pers						NLS
Compugraphic 8600		UNI.C			Arbor Pers	Wash St		U Wisc			NLS
Compugraphic 8800					Arbor						
Harris 7500										SARA	
Hell Digiset				GMD2			GMD2				

Why TEX Should NOT Output PostScript — Yet: Addendum

Shane Dunne
University of Western Ontario

[Editor's note: The original article appeared in Volume 9, No. 1, pp. 37–39. The following addendum was received after the issue had been sent to the printer.]

Insert at the end of the last full paragraph, page 38. An example of such a precise specification can be found in the documentation for the X11 window system under development at M.I.T. [3]

Additional Reference:

[3] Gettys, J., Newman, R., and Schiefler, R.W. *Xlib: C Language X Interface Protocol Version 11.* Massachusetts Institute of Technology, Laboratory of Computer Science, Cambridge, Massachusetts, 1987.

ASCII Preview with vuTEX

Warren Wolfe

The call for an ASCII previewer for TEX (Brown, vol.9, no.1, 1988) prompts me to report the release of just such a program. Created to reduce the *edit* – TEX – *print* cycle and the associated costs in time and laser produced output, the program has been tested and modified since January, 1988, and has proven to be an effective tool in our TEX treasury. As the output device is nonspecific, *i.e., any ASCII device*, we have abandoned the usual dvigen name format and have given the program the name vuTEX.

Features

vuTEX was developed from Rokicki's dvigen model in a naive attempt to satisfy our own immediate needs for a previewer. Thus, the design model varies from that proposed by Brown, but nonetheless is justified in the manner in which it satisfies our own design criteria:

1. Words, lines, paragraphs, and pages must appear as integral units as produced by TEX.
2. Characters from special fonts and symbols should be represented in a meaningful way by ASCII characters.
3. Alignments, equations, and tables should be reproducible.
4. The program should be fast and easy to use.
5. The output should be directed to any ASCII device or to a file for screen editor viewing.

The result is a stripped down and rebuilt dvigen model in which many of the parts have the same name and intended function, but the way the job is done has been altered dramatically. The limited character set and resolution of the simplest ASCII device were chosen as the limits on the resources available to vuTEX in its generic form, but it is adaptable to use any special abilities of particular devices, such as overstrike and reverse imaging. vuTEX offers the following features:

- Input is from a standard dvi file and font tfm files.
- Output is printable ASCII code which may be addressed to a file (for screen editor viewing) or to a device such as a terminal or lineprinter.
- Fonts of all sizes and most styles are simulated. Many special characters are mapped onto ASCII characters which hint at their true meaning, e.g. S for $\int$. Ligatures are represented by the group of characters so that ffl appears as `ffl`. Unrepresented characters are replaced with `#`.
- Used fonts are ranked so that questions of overlays or overstrikes are resolved by priority.
- The process is fast. Typical output on a Honeywell DPS8 is 1024 processed characters per second with a 60 page, 167,000 character, document processed to a file in less than 3 minutes.
- Horizontal spacing may be selected to retain the vertical alignment produced by TEX (for tables and equations) or to compress interword spacing (for ordinary text).
- Sub/superscripts may appear on different lines than the base line.
- Output may be truncated at the right margin. The coarse resolution of the fixed pitch font results in output that is wider than the 80 characters on a typical terminal screen. To avoid wraparound, the user can select the width of the printed output. Some terminals, and most printers, will print 132 characters to a line while many screen editors allow viewing of wide records with single key operations. Thus, vuTEX avoids the left-right-centre views proposed by Brown.

- The source for vuTEX is webbed Pascal and is expandable and adaptable to various systems. Current versions are available for Honeywell CP6 and **unix** systems.

Overview of vuTEX

This section is intended to rationalize the limitations of vuTEX and to describe more fully some of the options available.

1. Initialization and Dialog. During the initialization, several default parameters are set. These may be altered by the user through command line input or a brief interaction with the program and include:

- starting page number;
- number of pages to be printed;
- print width, *i.e.*, the maximum number of characters across a line before truncation occurs;
- compression mode (as explained below).

The program normally will run silently and, if the output is to a device other than the terminal, only the processed page numbers will appear on the screen.

2. Prescan. As in many dvi drivers, vuTEX makes two passes over the input file. In the prescan, all character and font usage is recorded. It is noted that, in this implementation, \special is ignored.

3. Font Loading. The fonts used in the file are ranked according to their printability and the frequency of usage. The font with the highest priority is used to determine the horizontal resolution *i.e.*, the number of dvi units per single space on the output device. The vertical resolution (dvi units per line) is calculated to place a maximum number of lines on a page. Although the character resolution is coarse, vuTEX maintains the accuracy of a dual system of dvi units in all positioning.

4. Page Construction. During the second pass, vuTEX skips over any undesirable pages and creates each printed page separately. According to its font type, each character is mapped to an ASCII character which is positioned in a word. A word is a sequence of characters with no substantial horizontal or vertical space between elements. TEX provides a certain width of space (in dvi units) for the word, and vuTEX places the characters as left justified in that space. Spaces do not appear in words because of wider or narrower characters from proportional fonts. If a word consists of characters from a small pitch font, the dvi space allocated might not transform to a space on the output device that is wide enough to hold all the characters. Thus, such words may appear chopped or with omissions.

Moderately large vertical motions in a line are interpreted as the creation of sub or superscripts. A new word is begun, usually on a new line. Indeed, there may be subscripts to subscripts to superscripts with an initial base line.

Once a word is complete, it is positioned on a page array so that the leftmost character of the word is mapped onto its transformed dvi coordinates on the page. A companion priority array determines questions of overlays.

Vertical and horizontal lines are constructed from | (bars) and — (dashes) respectively, and are given low priority so that more meaningful characters will not be overwritten.

Note that there is no assumption about the sequence of characters in the dvi file. The vuTEX "pen" can go back and fill in blank spaces on a page. If the coordinates of a word or character are beyond the edges of the page array, truncation will occur. The top and left margins have offsets which may be used for characters pushed into these spaces.

5. Page Printing. Once a page is complete, the array will consist of lines of words, often with several spaces between them. Most sub/superscripts will appear on separate lines with a mapping array determining which are baselines and what lines appear where. Typically, only lines with nonblank characters will be printed, but, if large blank vertical spaces are to appear (for inserts or graphs), vuTEX will attempt to display this by printing an appropriate number of blank lines.

If the user selects the true TEX mode, the page array is printed exactly as it was constructed. Vertical alignment of words is as determined by the dvi file, but characters in a word are left justified in the space provided so that gaps may appear between words. This mode is desirable when previewing tables or equations, or when sub/superscripts are to be retained. Example 1 demonstrates the output of vuTEX in this mode.

If the compress mode is selected, an effort is made to eliminate blank space so that the output is narrower and a sense of the proper ratio of black type to white space is presented. First, all associated sub/superscripts are squeezed into available space in the base line. Although this may be modified later to preserve the off-the-line printing, vuTEX's current algorithm is unable to drag the associated sub/superscript along with the characters in the base line as shifts are made to close the gaps. Thus, lines are compressed vertically first so that off-the-line script may appear in the base line beside the associated character or word. During horizontal compression, the first word in each line retains its original spacing in the line so that left indentation is maintained. All other characters are shifted left so that a single space appears between words. Of course, this destroys most vertical alignment, and deliberate horizontal spacing disappears. However, the effect is often a more readable version, particularly with ordinary text. See Example 2.

Finally, the page array is printed. Depending on the page width selected by the user, the output may be truncated at the right margin. The first column is a key column which may contain:

+ if the line is a superscript to a lower line
− if the line is a subscript to a higher line
∗ if the line contains characters from fonts other than the priority font
> if characters are missing from the line due to truncation or overlays.

A graded horizontal rule delimits each page or, for systems that support such, a top of form might be sent.

Conclusion

vuTEX makes the preparation of TEX documents easy and inexpensive. Requirements for special graphics terminals or workstations are reduced and every user on the system has access to a previewer. Inquiries should be addressed to the author at the Department of Mathematics, Royal Roads Military College, Victoria, B.C., Canada.

```
 Example       1.

*         EXERCISE8.11

*                    S #
*                          t - ib
+                         ________ iat        ab
+                                 e    dt =  e  E 1(ab),        a,b >  0.
+                          2     2
*                     0   t  + b


          Page    168  of  the  T  Xbook.
-                                E


 Example 2.

 English words like ‘technology’ stem from a Greek root beginning with the lette
*###...; and this same Greek word means art as well as technology. Hence the
*name TE X, which is an upper case form of ###.

*Insiders pronounce the # of TE X as a Greek chi, not as an ‘x’, so that TE X
*rhymes with the word blecchhh. It’s the ‘ch’ sound in Scottish words like loch
*or German words like ach; it’s a Spanish ‘j’ and a Russian ‘kh’. When you say
 it correctly to your computer, the terminal may become slightly moist.

 Page 1 of the TE Xbook.
```

Site Reports

Macintosh Site Report (*Textures*)

Barry Smith
Blue Sky Research

Editor's note: Owing to an unfortunate error, this report was omitted from the last issue of TUGboat. The news, though no longer new, is still important to users of *Textures*. And a later development has found Kellerman & Smith ending their joint venture and forming two separate companies. Barry Smith is now associated with Blue Sky Research, and David Kellerman, with Northlake Software. The text below has not been edited, so it should be read with these changes, and the delay, in mind.

We have two major pieces of news this month.

First, Addison-Wesley has decided to withdraw from the TeX software business. Effective immediately, we at Kellerman and Smith will be assuming an active role as the publisher of *Textures*. We believe this will result in better, faster service to *Textures* users. We have installed a direct-access *Textures* telephone line, 800-622-8398 — please feel free to call if you have any questions or concerns we can help resolve.

Second, we have released a new version of *Textures*, version 1.01. This includes hundreds of small improvements, and fixes a printing problem introduced by Apple in the new System 5.0 release. *Textures* 1.01 is faster at typesetting, twice as fast at screen preview, and now can run TeX in the background under the new MultiFinder system.

To the best of our ability, we have directly notified all registered *Textures* users of the availability of *Textures* 1.01. If you are a *Textures* user and have not heard from us already, please give us a call. In particular, if you have never returned a registration card, you should let us know that you exist and would like to be kept informed of updates. We apologize for any confusion caused by the transfer of publishers.

TeX on the Cray

Bart Childs

TeX is now available on the Cray. I have done this implementation under the CTSS operating system using the old straightforward method of creating a WEB change file. The Cray Pascal is pretty good, even though it has a few restrictions. I assume that it will be easiest to install it on other Cray operating systems with another link step. It seems that many Cray systems never bothered purchasing Pascal.

There were two big problems:

1. finding the right system calls to enable avoiding the Cray record structure (absolute I/O had to be used), and
2. coding our way around the flat file structure that has case sensitive variable names with a maximum of eight characters.

The Cray implementation is working at Los Alamos and has been sent to Lawrence Livermore. By the time this is in print, we should have a stable package ready for public distribution.

A more detailed paper will be available soon that explains the design decisions that were made.

Many people probably wonder, why bother implementing TeX on a Cray? There is a simple, logical answer. Institutions that have a large number of professionals whose primary computing facility is a Cray, don't need these professionals learning another system interface. I found the CTSS interface to be about as oppressive as those in *unix* and MS-DOS. The flat file structure is much worse.

I will serve as the Cray site coordinator unless someone else wants the job and informs us before the next meeting of the steering committee.

Data General Site Report

Bart Childs

We have updated to TeX 2.9 and our standard distribution now uses the 64 bit TeX discussed in another article (see page 122). This is faster on systems that are not starved for memory. The output driver table has now been corrected; my apologies to Don Hosek for being so slow in keeping him posted.

Warnings & Limitations

Controlling <ctrl-M>; Ruling the Depths

Barbara Beeton

Two problems have recently surfaced to snare some unwary users. One (the easy one) has to do with how TEX handles the depths of rules in the absence of explicit instructions. The other, a more insidious problem, is what happens when TEX encounters a "bare" control-M in its input stream. Let's take the nasty one first.

Beware of the bare <ctrl-M>

We received a phone call in which it was pointed out that some sections of text were missing from a published paper for which the author had created the input and the camera copy was prepared at AMS, An investigation uncovered the fact that in some lines of text, a "bare" carriage return, or `<ctrl-M>`, occurred in the middle, in fact, immediately before the missing text. This was the condition of the file when we received it, and the file had not been changed subsequently. However, nothing had been missing on the proof, which had been prepared by the author.

Something had happened that is not supposed to happen with TEX: the same file, run through two different implementations of TEX, had produced different results.

We demonstrated the system-specific nature of this problem by constructing the following test file and running it through three different implementations of TEX available to us:

```
The quick brown^Mfox jumped over the
lazy dog.
```

On a DEC-20/TOPS-20, using the Stanford implementation, the output read "The quick brown fox jumped over the lazy dog." On a VAX/VMS, both the Stanford and the Kellerman & Smith implementations yielded "The quick brown lazy dog."

While talking with a colleague who uses a VAX/Unix Pastel implementation of TEX, I mentioned the problem and he ran the test through his system. This yielded a third result: an error message,

```
! Text line contains an invalid character.
```

The TEXbook, page 343, declares that `\catcode`\^^M=5`, which on page 37 is interpreted as "end of line". On the systems with which I am most familiar (DEC-20/TOPS-20 and VAX/VMS), the traditional end-of-line marker is an ASCII carriage-return/line-feed pair (⟨*crlf*⟩ or `^M^J`). In files I have received from Mactexintosh users, the bare `^M` seems to be the norm. (TEXing an unaltered Mac file on a DEC-20 invariably exceeds the input buffer, so I had grown accustomed to translating Mac `^M`s to ⟨*crlf*⟩s on receipt of a file, if the author had not already made the conversion.) And I understand that some other systems prefer record-oriented input, with no explicit end-of-line marker.

It is thus apparent that there are several interpretations to the bare `^M`, and they seem to be operating system dependent, or at least implementation dependent.

- The Stanford TOPS-20 implementation ends a line and assumes that the character which follows starts a new one; in other words, `^M` by itself, although nonstandard, is equivalent to ⟨*crlf*⟩.
- The Stanford and K&S VAX/VMS implementations end consideration of the content of the line (treating the `^M` the same as a `%`) and skip to the next ⟨*crlf*⟩.
- The Unix Pastel implementation does not permit a bare `^M`, only a ⟨*crlf*⟩ pair.
- There may be others — this investigation has really only just begun.

I inquired about this varied behavior, and was informed by David Fuchs that it was really a user problem, attributable to incorrect transfer of the file from one system to another. He said,

> Different run-time I/O systems for different compilers work differently, and there's nothing we can do about it. ... TEX is line-oriented; when moving TEX files around, you must choose a technique that preserves lines in the respective native modes of the various machines you are moving between. On various different systems this may mean mapping between CR, CRLF, LF, or padding out to a fixed record length (80, for instance) while dropping any CR/LF, or even putting a byte-count at the start of a variable-length record that may or may not need a CR/LF. Whatever is the native mode of the system is exactly what is correct; if you just carelessly transfer bits, you're out of luck!

Asking for further clarification from Don Knuth, I received the following reply.

> This is not a bug in TEX, since different installations do support different character

> sets. There are two ways to make a TEX file machine dependent, both mentioned in the manual: (1) Name your files with local conventions that use something other than letters and digits. (2) Use non-printing ASCII characters in your source files.
>
> ...
>
> Some installations of TEX will accept TAB characters (and treat them as spaces); others will not. ... See *TEX: The Program*, section 23; the *xchr* and *xord* can be different on different computers.
>
> ...
>
> In other words, TEX is working just as we designed it. The users who expect identical behavior across machines have to abide by (1) and (2).

I'm still not happy, because there is still a problem. More and more TEX users are shipping files around to other sites, where they are to be TEXed to generate copy for publication, or read, perhaps changed, and shipped on somewhere else. Not all these locations will have identical hardware and software, and, given the opportunity, Murphy's Law always applies.

I don't have any good solution to this problem. The only (bad) solution I can think of is, if your system is one that handles this situation poorly, you should check every file for bare `^M`s and fix them up before submitting the file to TEX. I would welcome a discussion of this problem by the TEX implementors, since, although it appears to be a communications problem between disparate systems, it does seem to bend the "same input, same output" principle that we've come to expect TEX to follow.

Rules can be a deep subject

One of my correspondents sent me an interesting puzzle a while ago. In what he thought was a simple use of a "fill-in" rule in an `\halign`, he suddenly found that the rules varied in thickness. Here is the preamble he was using:

```
\halign{# \hfill
   &# \vrule width 12pc height .5pt \cr
... }
```

and this is what he got:

Name: ______________________
Address: ______________________
____
Telephone: ______________________
Best time to call ______________________

The rule on the next-to-last line has taken on the depth of the "p".

This is what he wanted:

Name: ______________________
Address: ______________________
____
Telephone: ______________________
Best time to call ______________________

I can see two possible solutions.

- Explicitly specify the depth; otherwise the "environment" depth will be used. In the "corrected" version above, `depth 0pt` was added.
- Use leaders instead of a `\vrule`: `\leaders\hrule\hskip 15pc` gives the same result as the `\vrule`.

Actually, my correspondent had tried an `\hrule` first, but got the nasty message that leaders were required in that context. Remembering that one must always nest opposites (`\vrule`s in horizontal mode, `\hrule`s in vertical), and being too lazy to look up how to use leaders, he simply switched to the `\vrule`. A reasonable approach. But there are a few things that one must remember not to take for granted.

Macros

German TEX, a Next Step

Peter Breitenlohner

1 The Present State

As reported by Joachim Lammarsch (TUGboat 8(1987)304) and described in detail by Hubert Partl (TUGboat 9(1988)70–72) the German TEX Users Group has agreed on a standard for a "Minimal Subset of German TEX Commands" at its 6th meeting in Münster (Germany) last October. This standard was, in fact, designed to a large extent by Hubert Partl and the present implementation in terms of TEX-macros is almost entirely his work.

The basic idea is old: make " an active character and define " as a macro with one argument such that the macro expansion yields whatever is necessary.

The present implementation has the advantage of high portability since TeX, TFM files, (German) hyphenation patterns and GF/PK/PXL files all remain unchanged. The only thing needed is a file GERMAN.TEX or GERMAN.STY which contains the necessary macro definitions. In particular " is defined to yield something like:

```
"s  →  \char25
"a  →  \accent\char127 a\allowhyphens
"c  →  \discretionary
         {k-}{}{c}\allowhyphens
"t  →  \discretionary
         {tt-}{}{t}\allowhyphens
"|  →  \discretionary
         {-}{}{\kern.03em}\allowhyphens
```

where o and u are treated like a (umlauts) and f, l, m, n and others are treated like t (special German \discretionarys). This solution has, however, several disadvantages. First the \accents and \discretionarys inhibit automatic hyphenation of the full word, only the parts before and (due to \allowhyphens) after the \accent will be subject to TeX's hyphenation algorithm. The "Umlaut accent" of the computer modern fonts does not yield a really satisfactory German umlaut, not even after some vertical shifting as applied in Partl's macro definitions; ideally there should be single characters, specifically designed for German umlauts, at some character position above 128 and accessed via ligatures. Finally in many areas (e.g., physics, mathematics, ...) we actually live in a bilingual environement. Many publications (like the one you are reading in this moment) are written in English, other publications and most letters are in German. Ideally something like Ferguson's multilingual TeX (see e.g., TUGboat 6(1985)57–58) should be available for English-German-French-Italian-...

In this article I want to discuss what can be done with modifying the TFM files and the hyphenation patterns, but nothing else. This still guarantees a reasonable amount of portability and is similar to the approach for "French in TeX" described by Alonzo Gariepy (TUGboat 9(1988)65–69). The basic strategy can already be found in Bernd Schulze's article in TUGboat 5(1984)103–104 and he also has created German hyphenation patterns with umlauts.

I hope some day there will be fonts with genuine German umlaut characters, but then it will need some more time until such fonts are available for all kinds of output devices. Moreover I have a strong suspicion that many of the existing device drivers do not handle the characters 128–255 correctly.

As far as multilingual TeX is concerned, it will be a long time until it will (perhaps) be available for all kinds of computers. To make multilingual TeX available as public domain software would certainly help this process. At present I see no point to invest any work in this direction.

2 The Second Step

Let me now discuss in detail how German TeX can make use of kerns and ligatures. These ligatures are not, as for the French accents, straightforward ones but they are mostly artificial "trick ligatures". In order to work they must use (abuse) two characters which are (at least in PLAIN TeX and in LaTeX) never accessed directly but only through control sequences. For the moment let us denote them by α and β.

2.1 The Macros

First Partl's macro definitions are modified to yield:

```
"a  →  aαa
"c  →  cββ
"t  →  ttα
"|  →  β
```

and similar for other characters. The α and β in these definitions are two suitable characters with category code 11 (letter) or 12 (other character). It is important that the above macro expansions involve nothing other than expansion of expandable tokens in order to use them in the hyphenation patterns. The macro definition may, however, start with \relax (or with LaTeX's \protect) provided we define \let\relax=\empty while reading the hyphenation patterns.

2.2 The Hyphenation Patterns

The first part of the hyphenation patterns is made by hand and contains:

```
a8α8a % never hyphenate inside "a
c8k % never hyphenate inside ck
8c8k9k8 % hyphenate ckk as ck-k
8c8β9β8 % hyphenate "ck as cβ-βk
8t8t9t8 % hyphenate ttt as tt-t
8t8t9α8 % hyphenate "tt as tt-αt
8t8z9z8 % hyphenate tzz as tz-z
f9β8 % hyphenate f"|l as f-βl
```

and similar for other characters. These patterns should in fact be \input from a separate file.

The remaining patterns are as usual and may contain "a, "o, "u and "s which will be converted into aαa and so on. One can in fact adapt Bern Schulze's file GHYPHENU.TEX for this purpose, but I hope that new and more efficient patterns will be available in the near future.

2.3 The TFM files

The ligatures and kerning in the TFM files (but nothing else) must be modified. First we add ligatures:

aα → ¨

cβ → k   kβ → c βk → k

tα → α αt → t

βf → f βi → i βl → l

with the effect that:

"a → ¨a

"ck → ck or k-k

"tt → tt or tt-t

f"|l → fl or f-l

Note that the kerning between e.g., "a and the adjacent characters is the same as between a and the same adjacent characters. This will in general (although not in all cases) be just what is desired.

As a last step we introduce a small positive kern between f (or the ff ligature) and β as well as a negative kern between ¨ and either a or o or u in order to obtain e.g.,

"a → ä

f"|l → fl or f-l

This kerning works nicely for "a and "o since the umlaut accent has the same width as a and o. The u is, however, somewhat wider, and the placement is somewhat different than what TEX's \accent would produce. In my opinion this is a tolerable price for the full automatic hyphenation achieved this way.

Without modifying pixel files the only alternative would be to convert "u into the sequence u \kern ¨\kern u such that the accent is placed as desired and the two u's are on exactly the same position. With DVItype's pixel rounding mechanism this leads, however, to the possibility that the two u's are off by one pixel.

Note that the mechanism described above works only for lower case umlaut characters. For the upper case ones one has to rely on TEX's \accent or something similar. This seems tolerable since the remaining part of the word (after the initial character) will still be hyphenated automatically.

These modifications of the TFM files will be done automatically by a suitable program which is under developement. The resulting files should certainly be named differently than the original ones. Deviating from the French solution I propose not to change the file name but rather the extension, e.g., from .tfm to .grtfm or .gfm. For all the non text fonts (such as math italic, math symbol and others) these .gfm files would be identical to their .tfm counterparts. Supplying INITEX with a modified POOL file (just replace the string .tfm by .gfm) results in the effect that INITEX reads .gfm files. This will also be true for TEX once the .fmt file produced by INITEX has been digested.

2.4 The Choice of α and β

The use of α and β in the trick ligatures for German hyphenation should not interfere with the normal use of these two characters. We propose to use the two character codes 0 and 1, normally accessed through the control sequences \Gamma and \Delta. Since both \Gamma and \Delta are defined as \mathchar they will never be part of a ligature nor will they be affected by (implicit) kerning. Therefore the definition of these two control sequences need not be modified; this was the reason to choose them instead of e.g., $ or #.

Finally we have to assign non vanishing \lccodes to the two characters α and β in order that TEX accepts them as part of hyphenatable words. When this scheme was designed, I was under the erroneous impression that the \lccode for α and β must be equal to the character code. This would have prevented the use of character codes 0 (vanishing \lccode) and 1 (a bug in TEX does not allow \lccode=1 in the hyphenation patterns) for α and β, instead one could have used character codes 2 (\Theta) and 3 (\Lambda). Meanwhile Professor Knuth has pointed out to me that the \lccodes need not and in fact should not be equal to the character codes. Instead they should be close to the character codes for the lower case letters a...z in order that the hyphenation patterns can be stored and retrieved efficiently, i.e., we can define:

```
\lccode0=95
\lccode1=96
```

3 Conclusion

The implementation of "German TEX" described here will be achieved with a moderate amount of work (both man power and CPU time). It has the advantage that the resulting DVI file can be processed with the standard computer modern pixel files and that almost all words can be automatically hyphenated. The disadvantage is clearly that the shape of the resulting umlaut characters is a not fully satisfactory compromise.

Once the whole program is completed experience will teach us the weak points of the present design and how to eliminate them.

Some Problems with the INRSTeX Table Making Macros

Michael J. Wichura
University of Chicago

The INRSTeX table making macros distributed with TeXniques Number 2 greatly simplify the task of making ruled tables in TeX. There are, however, some (admittedly uncommon) circumstances under which the macros don't work as advertised or have adverse consequences. This article brings to light several such problems and suggests ways to correct them. Be forewarned that the issues here are fairly TeXnical — the kind of material that is marked with dangerous bends in *The TeXbook*.

1 A problem with \left, \right, and \-

The INRSTeX macros break what I would propose as the first commandment for authors of macro packages: *Thou shalt not redefine a TeX primitive.* \left and \right are TeX primitives that are used to make variable size delimiters in math mode. Within the table environment, however, INRSTeX preempts the meanings of \left and \right to control the positioning of items within columns. The original meanings of \left and \right are gone, and macros, such as Plain TeX's \big, \bigg, \Big, and \Bigg, that rely on those meanings being in force won't function properly. You'd be in for 'big' trouble if you tried to make a table whose entries involved some heavy math.

There's a similar problem with \-, which INRSTeX uses to draw horizontal lines in tables. \- is TeX's primitive for a discretionary hyphen. If you planned to make a table having (presumably narrow) paragraphs as entries, you might well want to use \-'s to assist TeX in hyphenation.

These difficulties wouldn't exist if INRSTeX had chosen slightly different names, say '\Left', '\Right', and '\Hr', for its commands. Name changing is not a feasible option at this point because the macros are already widely used. The macros can and should be supplemented by a command which makes the original meanings of \left, \right, and \- available within the table making environment. The command \restoreTeXprimitives (\rTp for short) defined below does just that.

```
\let\T@Xleft=\left
\let\T@Xright=\right
\let\T@Xdiscretionaryhyphen=\-
\def\restoreTeXprimitives{%
  \let\left=\T@Xleft
  \let\right=\T@Xright
  \let\-=\T@Xdiscretionaryhyphen}
\let\rTp=\restoreTeXprimitives
```

This code should be inserted at the end of the INRSTeX macros, just before the line that reads '\catcode\@=12'. With this code in place, you could, for example, typeset the expression

$$\left(\frac{a-1}{b-1}\right)$$

in a column with a display-style math template by entering '\rTp \left({a-1\over b-1} \right)'. It's not necessary to enclose such usages of \rTp in a group ({...}), since an INRSTeX table is created with an \halign and since TeX automatically enters an additional level of grouping when it works on each individual entry to an \halign.

There's a related problem concerning | and \|, which Plain TeX uses as delimiters in math mode, but which INRSTeX preempts to draw vertical rules in tables. One could augment the definition of \restoreTeXprimitives to cover | and \| as well, but there's a good reason not to do so. The table making macros set things up so that | and \| signal the end of a data column. If you were to restore the original meanings of these control sequences with \rTp, then you would have to enclose every usage of \rTp in a group. That's a greater burden than having to use Plain TeX's synonym \vert for |, and \Vert for \|.

2 Problems with \zerocenteredbox

The command \zerocenteredbox (\zb for short) centers its argument vertically in a box of zero height and depth. INRSTeX provides this feature as a means of doing some makeshift vertical spanning within tables. The manual asserts that \zb works correctly "even with display math templates". This, however, is not the case. The problem arises because display math templates have the form $\displaystyle{#}$ (and so should more properly be called display-style math templates), whereas \zb uses an \if type test for display math mode (what you get between pairs of $$'s). There is, in fact, no \if type test for display style. One has to use, instead, a \mathchoice or \mathpalette construction, as explained on page 151 of *The TeXbook*. The following code (re-)defines \zb correctly, in the way Plain TeX defines \phantoms and \smashes.

```
\def\zerocenteredbox{%
  \relax
  \ifmmode
    \expandafter\mathpalette
    \expandafter\m@thzb
  \else
    \expandafter\m@kezb
  \fi}
\def\m@kezb#1{%
  \setbox\z@=\hbox{#1}%
  \fin@shzb}
\def\m@thzb#1#2{%
  \setbox\z@=\hbox{$\m@th#1{#2}$}%
  \fin@shzb}
\def\fin@shzb{%
  \vbox to\z@{\vss\box\z@\vss}}
```

The `\relax` on the second line of the code is important, for reasons explained on page 240 of *The TeXbook*. There's no such `\relax` in INRSTeX's definition of `\zb`, and that version of the macro malfunctions when it appears as the first token in a table entry for a column with a math- or display-style math template.

3 Problems with \modifystrut (alias \mst) and \sa

These very useful commands allow you to fine tune the vertical and horizontal spacing in a table through appropriate struts. The commands, however, don't work in math modes. The INRSTeX manual sidesteps this issue by using constructions like '`\mst{$\int$}{0pt}{3pt}`'. In a column with a math template it would be preferable to enter just '`\mst{\int}{0pt}{3pt}`'. The following macros (re-)define `\modifystrut` and `\sa` so that they work in math modes as well as in horizontal mode. The idea is to again use a `\smash`-type construction; the macros for `\mst` are a little more complicated because they have to take into account additional arguments (e.g., the '0pt' and '3pt' in the example above).

```
\def\modifystrut#1#2#3{%
  % #1 = original
  % #2 = add to height
  % #3 = add to depth
  \relax
  \ifmmode
    \def\next{%
    \mathchoice
    {\m@thmst\displaystyle{#1}{#2}{#3}}
    {\m@thmst\textstyle{#1}{#2}{#3}}
    {\m@thmst\scriptstyle{#1}{#2}{#3}}
    {\m@thmst\scriptscriptstyle
      {#1}{#2}{#3}}}
  \else
    \def\next{\m@kemst{#1}{#2}{#3}}%
  \fi
  \next}
\def\m@kemst#1#2#3{%
  \setbox\z@=\hbox{#1}%
  \fin@shmst{#2}{#3}}
\def\m@thmst#1#2#3#4{%
  \setbox\z@=\hbox{$\m@th#1{#2}$}%
  \fin@shmst{#3}{#4}}
\def\fin@shmst#1#2{%
  \dimen\z@=\ht\z@
  \advance\dimen\z@ by #1\relax
  \dimen\tw@=\dp\z@
  \advance\dimen\tw@ by #2\relax
  \vrule width\z@
    height\dimen\z@ depth\dimen\tw@}
\let\sa=\hphantom
```

The INRSTeX manual states that the first argument to `\mst` can "even be a duplication of the row, as long as the row contains no explicit & characters and excluding the commands `\br` and `\er`." This assertion is correct if you use just the simple `\left`, `\right`, and `\center` templates, and if, as in all the examples in the INRSTeX manual, `\mst` appears as an argument to `\br` or `\er`. Otherwise, the assertion may be false, since `\mst` doesn't take templates into account, and since the definitions of the active characters `|` and `"` involve &'s when INRSTeX is working between `\br` and `\er`.

4 A problem with \use

`\use{<number of data columns>}` is asserted to "merge the next `<number of data columns>` into one and use the format or template of the last one." And so it does, unless `<number of data columns>` is one: '`\use{1}`' results in an error message from TeX. According to the intended use of `\use`, '`\use{1}`' should be the same as ⟨null⟩. To achieve this, `\use` should be (re-)defined as follows:

```
\def\use#1{%
  \ifnum #1>\@ne
    \omit
    \mscount=#1
    \advance\mscount by \m@ne
    \multiply\mscount by \tw@
  \loop
    \ifnum\mscount>\@ne
    \sp@n
  \repeat
  \span
\fi}
```

5 Problems with \thrule

The command \thrule{<height>} is supposed to insert a horizontal rule of thickness <height> across an entire data column. But if you enter, say,

```
... \thrule{2pt}|  ...
```

no rule is drawn; the rule *is* drawn if you enter

```
... \thrule{2pt}␣|  ...
```

This puzzling discrepancy arises because \thrule uses a \leaders construction, and because INRSTEX makes | an active character whose expansion begins with \unskip when INRSTEX is working between \br and \er. \unskip removes any glue item that immediately precedes it; this is what allows the INRSTEX manual to state that the command | "removes spaces to its left." (Remember that TEX treats two or more spaces the same as one space, and that a space is (normally) a glue item.) Leaders, however, are themselves a special kind of glue, so in the first example above the \unskip removes the leaders and no rule is drawn. By contrast, in the second example the \unskip removes just the ␣ and the rule is drawn. The way to solve this problem is to place an invisible non-glue item after the leaders, as in the following re-definition of \thrule:

```
\def\thrule##1{%
  \omit \leaders
  \hrule height ##1\hfill\null}
```

This will also fix similar problems that occur with constructions such as '\use{3} \-|'.

There is another problem with \thrule. When INRSTEX's \midtabglue is non-zero, the horizontal lines \thrule draws across data columns don't join up with vertical lines in neighboring rule columns, since the horizontal lines don't span the tabskip glue. Unfortunately, there's no easy way to modify \thrule so as to fix this problem.

6 A problem with \everycr

Within the table making environment, the INRSTEX macros set the \everycr token list to

```
\noalign {\global \a@lignstate=0}
```

Consequently, if you use an ordinary \halign within a table, you'll throw off INRSTEX's accounting and the table won't come out right. To get around this problem, you should use Plain TEX's \ialign in place of \halign; moreover, you should specify \normalbaselines (minimally) before the \ialign since INRSTEX turns off normal line spacing with \offinterlineskip.

7 A problem with \sp and \om

At the end of the INRSTEX macros, \sp and \om are \let equal to \span and \omit, respectively. This is not mentioned in the manual, nor are these abbreviations used anywhere in the macros themselves. Since they serve no useful purpose, they are best deleted.

8 A test table

When the INRSTEX macros are modified in the ways suggested above, this little contrived table

hyphen- ation	$\left[\dfrac{A+1}{B+1}\right]$
Here's an ialign	$\left[\dfrac{A+1}{B+1}\right]$

results from the following code:

```
\begintable
  \def\P{\vtop{\normalbaselines
    \hsize=.5in \raggedright \noindent
    \restoreTeXprimitives
    hy\-phen\-ation}}
  \def\PP{\P \mst{\P}{3pt}{3pt}}
  \def\M{\restoreTeXprimitives
    \left [ {A+1 \over  B+1} \right ]}
  \def\MM{\M \mst{\M}{20pt}{5pt}}
  \def\I{\vbox{\normalbaselines
    \ialign{##\hfil\cr
    Here's an\cr
    ialign\cr}}}
  \begintableformat
    \center " \displaymath\center
  \endtableformat
  \-
  \br{}             " \sa{\M\M} \er{}
  \br{|} \PP        |           \er{|}
  \br{|} \use{1} \-| \zb{\M}    \er{|}
  \br{|} \I         | \MM       \er{|}
  \-
\endtable
```

Box Plots and Scatter Plots with TeX Macros

A.J. Van Haagen

1 Introduction

The powerful macro facilities of the TeX language can be employed to create simple sets of instructions for drawing box plots and scatter plots. Both are important tools in the graphical analysis of statistical data.

Box plots, more fully known as box-and-whisker plots, were introduced by Tukey [2] as a graphical tool to give a summary of the distribution of a set of data by five numbers: the lower extreme, the first quartile, the median, the third quartile, and the upper extreme. An example of a box graph is given in Figure 1. The code for Figure 1 shows how easy it is to draw such plots with the set of definitions in the file `macboxplot.tex`. The figure shows how box plots can be used to compare batches of data. For a more detailed discussion of box plots see [3] and the references cited there.

Scatter plots serve many useful purposes in statistics. They are pervasive in regression analysis and can also be used effectively in areas like nonparametric statistics and Time Series Analysis. With the definitions in the file `macplot.tex` scatter plots and also some related graphs can be drawn very easily. An illustration is provided by Figure 2. The input has been computed by SAS [1].

2 Drawing Box Plots with TeX Macros

To draw box plots, first input the macros of the file `macboxplot.tex` with the `\input macboxplot.tex` command. This command is followed by the optional scale commands `\xscale[`*scalefactor*`]` and `\yscale[`*scalefactor*`]`. The entry *scalefactor* is an integer in the range $1, \ldots, 100$. The default value of the `\xscale` and `\yscale` command is 100. These commands can be used to scale the picture in horizontal and vertical direction, respectively.

The commands for drawing the box plots open with `\beginboxplot` and close with `\endboxplot`. The `\beginboxplot` command must be followed by the command `\range[`*low,high*`]`. The entries *low* and *high* should be integers greater than -2^{31}, but less than 2^{31}, such that $high - low \leq 214748$. The entry *low* corresponds with the bottom of the rectangular box in which the box plots are enclosed, the entry *high* with the top. The actual choice depends on the box plot parameters and will be discussed below.

Next comes the command `\ordinates[`*l,h*`][`*n,t,lpos,hpos*`]` which provides ordinates and tick marks. Here l and h are integers greater than -2^{30}, but less than 2^{30}, such that $l < h$ and $h - l \leq 214748$, n is the number of tick marks ($n \geq 2$), and t is 1,10,100,1000, or 10000. The parameters *lpos* and *hpos* are optional. They determine the positions of the first and last tick mark with their corresponding ordinate values, respectively. The values *lpos* and *hpos* must be integers between *low* and *high*. Their default values are *low* and *high*, respectively. For correct results $h - l$ should be a multiple of $n - 1$. The values l/t, $(l + i)/t$, $(l + 2i)/t$, $\ldots$, h/t, where $i = (h - l)/(n - 1)$, are placed to the left of the tick marks. Small tick marks are drawn half way between the ones just referred to.

A box plot is drawn by the command:

`\boxplot[`*boxplotlow,1st quartile,median,*
3rd quartile,boxplothigh,mean`],`

where *boxplotlow* is the lower extreme of the box plot and *boxplothigh* is the upper extreme. The entry *mean* is optional. The mean is indicated in the box plots by an asterisk. All entries are integers between *low* and *high*. To obtain these entries multiply the box plot parameters, as computed by SAS for instance, by a suitable power of 10 and round off or truncate. Convenient integer values for *low* and *high* are then selected.

The command `\boxplot` is followed by the optional commands `\outsiders[...]`, `\outlabelleft[...][...]`, and `\outlabelright[...][...]`. The number of box plots that can be drawn is limited only by the capacity of the TeX memory. They will be scaled and positioned automatically. The `\outsiders` command can handle any number of outside values, limited again only by the memory capacity of TeX. The command `\outlabelleft` writes a label to the left of an outside value, and `\outlabelright` writes one to the right.

After the box plots have been drawn there are the following options. The command `\boxplotlabels[...]` writes labels under the box plots. The labels may be numerical or may consist of text. The `\vertlabel[...]` command writes a vertical label to the left of the ordinates. Finally, the command `\text[...]` inserts text below the box plot labels.

The way these commands are used is evident from their appearance in Figure 1.

Since it is possible to scale down the picture to any desired size, one may wish the box graph to appear in the middle of text. The `\beginboxplot` ... `\endboxplot` sequence should then be enclosed by the TeX commands `\midinsert ... \endinsert`.

The macros in the `macboxplot.tex` file are compatible with the LaTeX system developed by Leslie Lamport.

3 Drawing Scatter Plots with TeX Macros

To draw a scatter plot, first input the macros of the file `macplot.tex` with the `\input macplot.tex` command. This command is followed by the optional scale commands `\xscale[...]` and `\yscale[...]` which are identical to the commands of the same name in the file `macboxplot.tex`.

The commands for drawing a scatter plot open with `\beginplot` and close with `\endplot`. The `\beginplot` command must be followed by the commands `\xrange[...]` and `\yrange[...]`. These commands set the lower and upper limits of the x coordinates and the y coordinates of the points to be plotted. The `\yrange` command is identical to `\range`. Next come the commands `\xaxis[...][...]` and `\yaxis[...][...]` which provide scales for the x axis and y axis. The command `\yaxis` is identical to `\ordinates`. Instead of these two commands, one can also apply the commands:

> `\xlabels[(`x_1,*xlabel1*`)(`x_2,*xlabel2*`),...]`,
> and
> `\ylabels[(`y_1,*ylabel1*`)(`y_2,*ylabel2*`),...]`.

The first entry within each pair of parentheses indicates the location on the x axis and y axis, respectively. It should be an integer between the parameters in `\xrange` and `\yrange`, respectively. The second entry is the label one wants to write near that location.

The following commands: `\points[...][...]`, `\leftlabel[...][...]`, `\rightlabel[...][...]`, `\horlabel[...]`, `\vertlabel[...]`, `\hordash[...]`, and `\vertdash[...]` can be applied in any order.

The `\points[`*plottingsymbol*`][(`x_1,y_1`)(`x_2,y_2`),...,(`x_n,y_n`)]` command places the plotting symbol in the locations given by the coordinate pairs. The plotting symbol can be almost any symbol in the TeX system.

The `\leftlabel[...][...]` command writes a label to the left of a point and `\rightlabel[...][...]` writes one to the right. The `\horlabel[...]` command writes a label below the x axis and `\vertlabel[...]` writes one to the left of the y axis. The commands `\hordash[...]` and `\vertdash[...]` draw horizontal and vertical lines of dashes, respectively. The last command is the `\text[...]` command which writes text below the horizontal label.

The use of most of the these commands is illustrated in Figure 2. The macros are again compatible with the LaTeX system.

Besides the TeX macros described above, there is a definition which requires PostScript for drawing slanted lines. It is of the form:

> `\PSpolyline[`*linethickness*`][(`x_1,y_1`)`
> `(`x_2,y_2`),...(`x_n,y_n`)]`.

The entry *linethickness* is a number which sets the thickness of the polygonal line connecting the points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$. The unit used by PostScript is the bp.

Acknowledgement. The author wishes to thank Professor L. Gordon for his suggestion to try to make TeX draw statistical graphs and for his constructive remarks. He also wishes to thank the department of mathematics of the University of Southern California for allowing him to use their excellent TeX facilities.

References

[1] SAS Institute Inc. *SAS/STAT Guide for Personal Computers, Version 6 Edition.* Cary, North Carolina, 1985.

[2] John W. Tukey. *Exploratory Data Analysis.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1977.

[3] Antonius J. Van Haagen. *Box Plots and Scatter Plots with TeX Macros.* A Thesis for the Degree of Master of Science in Statistics, University of Southern California, Los Angeles, 1987.

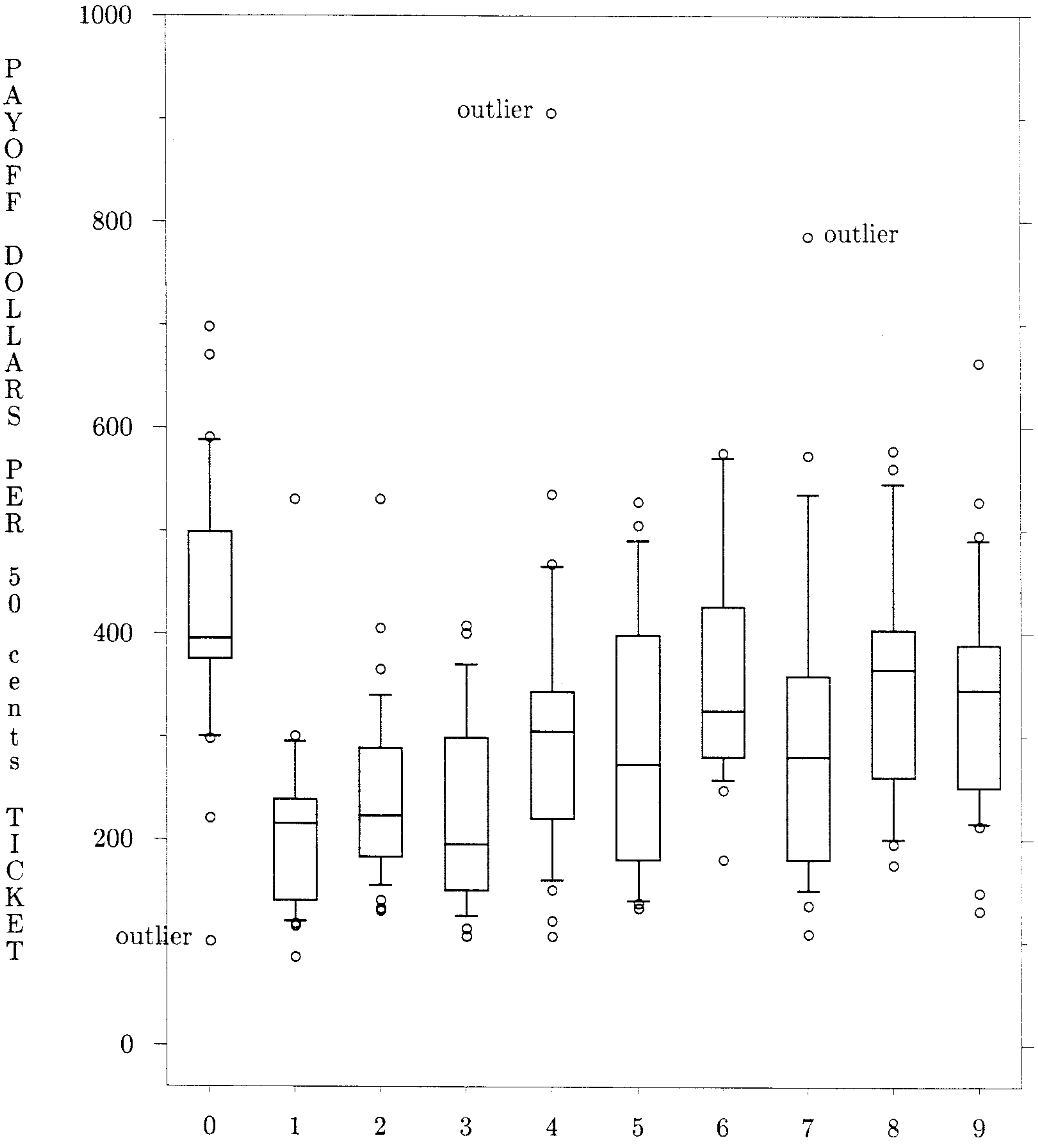


Figure 1: BOX GRAPH. The vertical scale is payoff of the New Jersey lottery, or numbers game, in which a player picks a three-digit number from 000 to 999. Winners share half of the pot. Each box graph shows the distribution of payoffs for all numbers with a particular leading digit. A leading digit of zero has the highest payoffs because fewer people tend to pick them. As the leading digit increases from one to nine the payoffs increase in a zigzag fashion, showing odd first digits are preferred to even. (From: The Elements of Graphing Data by William S.Cleveland. The data for this picture have been obtained by measuring the picture in Cleveland's book.)

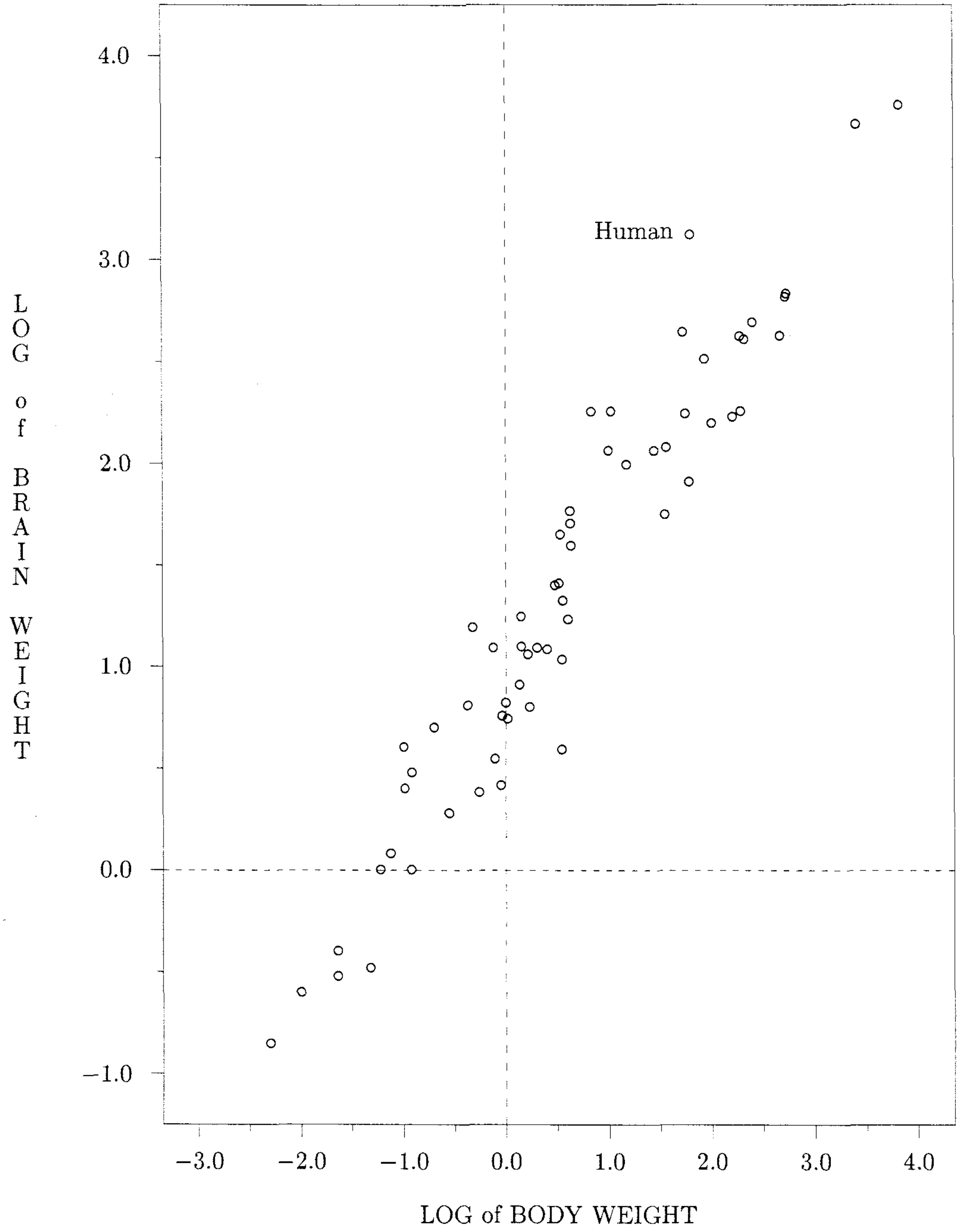


Figure 2: A scatter plot of brain weight data. The variable $log_{10}(brainweight)$ is plotted against the variable $log_{10}(bodyweight)$. Brain weight is the average brain weight in grams and body weight is the average body weight in kgs for 62 species of mammals.

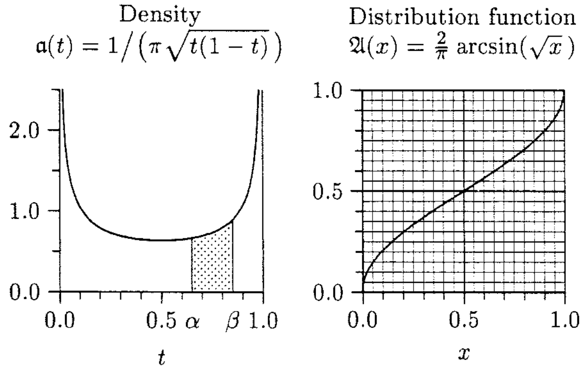
PICTEX: Macros for Drawing PICtures

Michael J. Wichura
University of Chicago

Overview

In the preface to *The TEXbook*, Knuth describes TEX as a "typesetting system intended for the creation of beautiful books — and especially for books that contain a lot of mathematics". PICTEX is a collection of TEX macros by means of which TEX users can easily instruct TEX to typeset beautiful pictures as a part of their books — and especially mathematical figures, such as the one below.

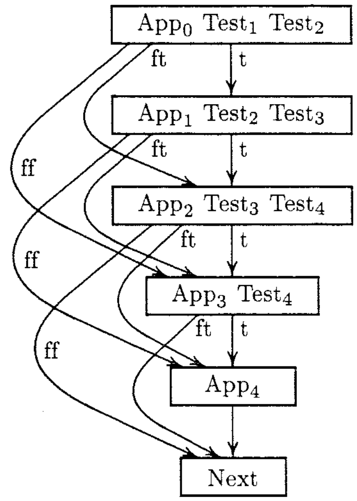
Figure 1. *Density $\mathfrak{a}$ and distribution function $\mathfrak{A}$ of the arc sine distribution. The shaded area under the graph of $\mathfrak{a}$ is $\mathfrak{A}(\beta) - \mathfrak{A}(\alpha)$.*



That figure and the others in this article illustrate the main things you can do with PICTEX: place text into a PICture; construct x and y axes with tick marks, tick labels, and axis labels; draw rectangles and other things made out of horizontal and vertical rules; draw straight lines and curves (without recourse to special fonts); use line fills that can be solid, dotted, dashed, or otherwise; and shade regions. In addition to these "primitive" graphics capabilities, PICTEX provides some "upper" level commands for drawing things like bar graphs, histograms, arrows, circles, and ellipses. Using TEX's powerful macro facilities, you can readily create other upper level commands that are tailored to your specific needs.

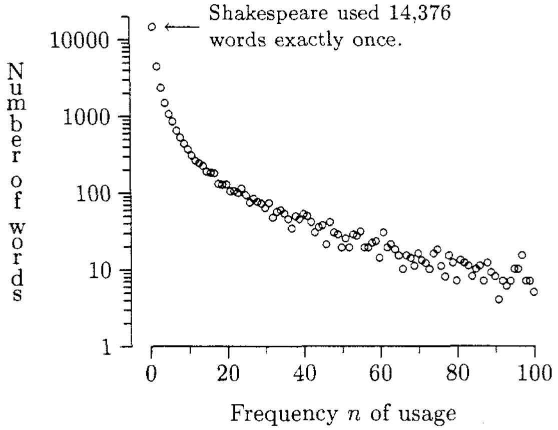
PICTEX has these advantages: (1) Figures become an integral part of the typesetting process. You can avoid having to leave the proper amount of space in your document for material that has to be created on some external device and later stripped into the finished product. (2) All of TEX's formatting capabilities are available for annotating your figures. In addition, that annotation will be done in the same fonts that you're using in the rest of your document. (3) Just as TEX is machine independent, so too is PICTEX. It doesn't matter whether you're working on a PC or mainframe computer. (4) Since typeset figures are embedded in the `dvi` file along with the rest of your document, all the advantages of TEX's device independent output accrue to them. In particular, you can revise away to your heart's content on your local system until your PICtures look just the way you want them to, and then you can have the final copy elegantly printed on a high resolution output device. (5) PICTEX can be extended using TEX's macro facilities, and can be used with LATEX.

Figure 2. *A segment of a complex flow chart.*


App$_0$ Test$_1$ Test$_2$
ft
t
App$_1$ Test$_2$ Test$_3$
ft
t
ff
App$_2$ Test$_3$ Test$_4$
ft
t
ff
App$_3$ Test$_4$
ft
t
ff
App$_4$
Next

On the other hand, PICTEX has several limitations: (1) PICTEX was expressly designed to facilitate the construction of pictures such as Figure 1. It simply is not the right tool for producing illustrations such as the lions that grace the title pages of *The TEXbook.* (2) Within the realm of mathematical figures, PICTEX itself doesn't make 3D pictures or other complex things. Considering that TEX provides fewer arithmetic capabilities than the simplest pocket calculator, that would be asking for too much. However PICTEX can be used as an interface between TEX and a sophisticated graphics program. For example, Figure 2 was set using Fig and a Fig-to-PICTEX converter developed by Micah Beck. (3) PICTEX takes a while to draw a PICture. Figure 1 initially took 30 seconds on a Sun 3/60, the bulk of the time going into producing the two curves. In subsequent drafts PICTEX used a special routine to replot the curves and did the whole PICture in 10 seconds. (By contrast, running on a Sun 3/60, TEX processes a page of "straight

Figure 3. *Number of words Shakespeare used exactly n times, for n = 1 to 100 by 1.*



text" in about 1 second.) (4) PiCtures take up a sizable amount of computer memory. A large PiCture with several curves will exceed the capacity of a standard version of TeX. This difficulty can be circumvented by using a larger version of TeX, such as that developed by Bart Childs et al. (See page 122.)

The PiCTeX manual contains detailed instructions on the use of the PiCTeX macros, with many examples and exercises, along with a command summary and index. The manual is about 90 pages long. Both the macros and the manual may be obtained from TUG.

The next section of this article has some examples showing how one goes about drawing a PiCture. The final section has an example showing how PiCTeX can be extended via TeX's macro facilities.

Drawing PiCtures

Table 1 lists the code that was used to construct Figure 3, in which the common logarithm of the number of words Shakespeare used exactly n times is graphed versus n for $n = 1, 2, \ldots, 100$. In tabular form the data are as follows:

n	$\nu_n =$ *number of words used n times*	$\lambda_n = \log_{10}(\nu_n)$
1	14,376	4.1576
2	4,343	3.6378
3	2,292	3.3602
⋮	⋮	⋮
99	7	1.1761
100	5	0.6990

Figure 3 has n on the horizontal (x) axis, and λ_n on the vertical (y) axis.

Table 1. *PiCTeX commands for Figure 3. (The line numbers are not part of the commands.)*

```
 1 \beginpicture
 2 \setcoordinatesystem units <.02in,.4in>
 3 \multiput {$\scriptstyle\circ$} at
 4    1 4.1576     2 3.6378     3 3.3602
 5    4 3.1652     5 3.0183     6 2.9227
    (additional coordinates omitted here)
 6   95 1.0000    96 1.0000    97 1.1761
 7   98 0.8451    99 0.8451   100 0.6990 /
 8 \put {$\longleftarrow
 9   \vcenter{\hsize=100pt \raggedright
10   \eightpoint \noindent
11   Shakespeare used 14,376 words exactly
12   once.}$} [l] <4pt,0pt> at 1 4.1576
13 \setplotarea x from 0 to 100,
14    y from 0 to 4.301
15 \axis bottom
16   label {Frequency $n$ of usage}
17   ticks numbered from 0 to 100 by 20
18         short unlabeled quantity 11 /
19 \axis left shiftedto x=-5
20   label {\stack
21     {N,u,m,b,e,r,,,o,f,,,w,o,r,d,s}}
22   ticks logged
23     numbered at 1 10 100 1000 10000 /
24     unlabeled short from 2 to 9 by 1
25       from 20 to 90 by 10
26       from 200 to 900 by 100
27       from 2000 to 9000 by 1000
28       at 20000 / /
29 \endpicture
```

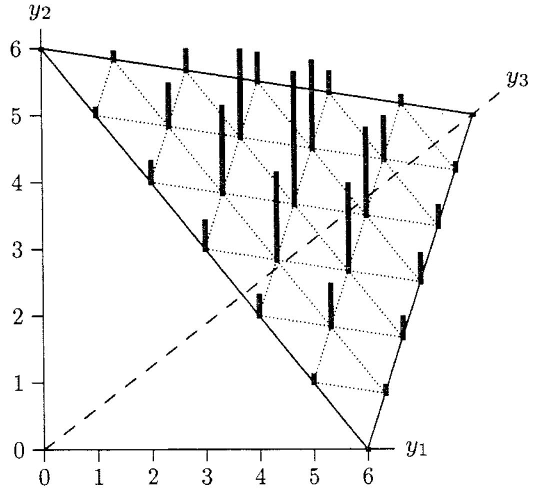
Table 1 is largely self-explanatory, since much of the code reads almost like English. However, some remarks are in order: (1) The command on line 2 tells PiCTeX to set up a rectangular coordinate system in which one unit on the x axis has length of .02 inches, and one unit on the y axis has a length of .4 inches. The value .02 was chosen so that a run of 100 units on the x axis would amount to $.02 \times 100 = 2$ inches, leaving plenty of room in *TUGboat*'s 3 inch column for the y axis structure. (2) The command on lines 3 – 7 places small circles at the coordinate points $(1, 4.1576)$, …, $(100, 0.6990)$. The trailing '/' on line 7 informs PiCTeX that the list of coordinate points is exhausted. (3) The option '`[l] <4pt,0pt>`' on line 12 to the `\put` command on line 8 instructs PiCTeX to place the associated text (i.e., the arrow and the `\vcenter`ed paragraph) into the PiCture with its left edge 4 points to the right of the the designated coordinate $(1, 4.1576)$.

(4) PiCTeX draws coordinate axes along the edges of a prespecified rectangular "plot area". The command on lines 13 – 14 sets up such a plot area, running from 0 to 100 along the x axis and from 0 to $4.301 = \log_{10}(20{,}000)$ along the y axis. The command on line 15 draws an axis along the bottom edge of this plot area. The analogous command on line 19 creates an axis along the left edge of the plot area; however this axis is subsequently shifted to lie along the line $x = -5$.

Some features of PiCTeX that emerged in the preceding example merit further discussion. The coordinate system is a key element of any PiCture, since it governs the placement of everything that's put into that PiCture. You can set and reset the coordinate system at will, moving the location of the origin and changing the lengths of the x and y units; this facilitates work on PiCtures like Figure 1 having several components. PiCTeX's `\put` command and its relatives have options that make it easy to specify exactly how an object should be positioned relative to a given coordinate. You can choose between various horizontal (centered, left, right) and vertical (centered, above, below, baseline) orientations, and you can offset objects horizontally and vertically by amounts that don't depend on the current coordinate system. `\axis` is PiCTeX's most versatile command. You can: choose between bottom, left, top, and right axes; freely specify where ticks are to be placed; use any combination of unlabeled, numbered, or user-labeled ticks; have the ticks marks point out from the plot area or into it, or even extend all the way across it; specify axis labels and a plot heading; govern the length and thickness of axes and tick marks; and adjust the spacing between the various components of the graph framework. In general you can fine tune any part of a PiCture; a few minor adjustments may make the difference between a figure that is merely presentable and one that is a work of art. One last point: you don't have to prespecify to TeX how much room a PiCture will take up. PiCTeX automatically determines the size of a PiCture and passes this information on to TeX so that the PiCture can be positioned appropriately in the page layout.

What about lines and curves? PiCTeX has four interpolation modes, two of which are piecewise linear interpolation and piecewise quadratic interpolation. (The other two modes generate histograms and bar graphs.) Moreover, PiCTeX has four modes for line fill: solid, dotted, dashed, and user-specified. Given a list of coordinate points, PiCTeX's `\plot` command connects those points using the current interpolation and line fill modes. Every option in PiCTeX has a default; the defaults for `\plot`ting are piecewise linear interpolation with solid line fill.

Figure 4. *Trinomial sample space and probabilities for $m = 6$ and $\pi = (1/3, 1/3, 1/3)$. For nonnegative integers y_1, y_2, and y_3 summing to m, the probability at the point (y_1, y_2, y_3) is $\frac{m!}{y_1!y_2!y_3!}\pi_1^{y_1}\pi_2^{y_2}\pi_3^{y_3}$. (The y_3 axis recedes into the plane of the page.)*



For example, in Figure 4 the edges of the simplex were drawn simply with

```
\plot   0 6   6 0   8 5   0 6   /
```

while the y_3 axis was created with

```
\setdashes
\plot   0 0   8.5 5.3125   /
```

In Figure 1 the left-half of the arc sine density $\mathfrak{a}$ was created by first placing the origin of the coordinate system at the point $t = 0.5$ on the horizontal axis, and by then entering

```
\setquadratic
\inboundscheckon
\plot    -.485 2.6187    -.475 2.0388
         -.465 1.7320    -.465 1.7320
         -.44  1.3403    -.40  1.0610
         -.36  0.9174    -.32  0.8285
         -.27  0.7564    -.22  0.7089
         -.12  0.6558      0   0.6366 /
```

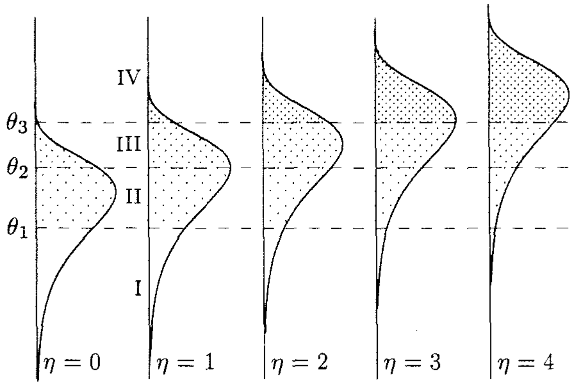
Note that $\mathfrak{a}(.5-.485) = 2.6187$, $\mathfrak{a}(.5-.475) = 2.0388$, and so on. The "in bounds check" feature kept PiCTeX from plotting any points with vertical coordinates greater than 2.5 . To plot the right half of $\mathfrak{a}$ the same commands were used, but with the signs of the horizontal coordinates changed to '+'.

In general, to `\plot` any given curve you have to provide PiCTeX with a list of coordinate values, as opposed to, say, some mathematical formula.

However, you can create such a list using Fortran or C or whatever, store it in a file, and have \plot take its input from that file.

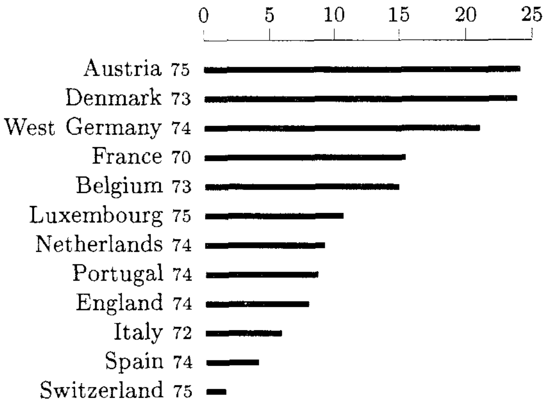
PiCTeX draws lines and curves by placing lots of periods close together. This takes a lot of time, because PiCTeX repeatedly has to lead TeX step by simple arithmetic step through complex calculations, like finding the distance between two points in a coordinate plane, for which it has no primitives. (TeX can only do fixed point addition, subtraction, and multiplication, and division by integers.) And all those periods take up a lot of room in TeX's memory. That's the down side; the up side is that there are no restrictions on the slope of lines and the curvature of arcs. By contrast, LaTeX's `picture` environment draws lines and circles by piecing together characters from specially designed fonts. This is fast and uses little storage space, but the choice of slopes and radii is quite limited. Fortunately, this is one instance where you can have your cake and eat it too. PiCTeX knows how to use LaTeX's picture objects, and PiCTeX's macros can be used inside LaTeX. (Don't try this without first reading the relevant section from the PiCTeX manual.)

Figure 5. *Diagram showing how the probabilities for the response categories* I, II, III, *and* IV *in a certain proportional hazards model vary with* η. *Probabilities are shown as shaded areas; higher numbered categories have greater shade density.*



PiCTeX shades a region by placing a "shading symbol" at every point of a "shading lattice" that falls within the region. You have control over both the symbol and lattice. In Figure 5 three different shading lattices were used, each with a \fiverm period as the shading symbol.

Figure 6. *Suicide rates in western Europe per 100,000 population per year for the years (19xx) indicated.*



Extending PiCTeX

Consider now how you might go about drawing Figure 6. To get started you could enter

```
\beginpicture
\setcoordinatesystem units <5pt,11pt>
\setplotarea x from 0 to 25,
  y from 0 to 0
\linethickness=.15pt
{\eightpoint \axis top
  ticks numbered from 0 to 25 by 5 / }%
\linethickness=2pt
```

These commands establish the coordinate system, draw the axis, and set PiCTeX's line thickness parameter to 2 points for drawing the black bars. The problem that needs addressing is how to place the bars and their associated labels into the PiCture. You could do this easily with PiCTeX's bar graph command, or even with TeX's \halign. However the point of this section is to illustrate a technique that can often be put to good use when there isn't an upper level command that does what you want to do. Let's agree then that you have to solve this exercise using just PiCTeX's primitive commands \put and \putrule. The most direct solution would be

```
\putrule from  0 -1  to  24.1 -1
\put {Austria \sevenrm 75}
  [Br] <-5pt,-2pt> at  0 -1
\putrule from  0 -2  to  23.8 -2
\put {Denmark \sevenrm 73}
  [Br] <-5pt,-2pt> at  0 -2
```

and a bunch of similar commands. (The 'Br's above stand for 'baseline right' orientation.)

It would be much less tedious to enter just the bare essentials, say in the form

```
\placebars
  Austria       5 24.1
```

```
Denmark      3 23.8
   ⋮
Switzerland  5  1.5 /
```

The question is: how should the macro `\placebars` be defined? Notice that the y coordinates of the bars decrease by 1 for each bar. This can be achieved by stepping a counter. It's okay to use `\count0` for this purpose, because PICture making goes on inside a group. The following definitions get the ball rolling:

```
 1 \def\placebars{%
 2   \count0 =0
 3   \Placebars}
 4 \def\Placebars #1 #2 #3 {%
 5   % #1=Country, #2=last digit of year,
 6   % #3=rate
 7   \advance \count0 by -1
 8   \putrule from  0 \the\count0  to
 9     #3 \the\count0
10   \put {#1 \sevenrm 7#2} [Br]
11     <-5pt,-2pt> at  0 \the\count0
12   \repeatifnecessary}
```

The line numbers aren't part of the macros; they're just for ease of reference. The `\Placebars` macro uses blanks to delimit its arguments, so you'll have to enter 'West Germany' as `West~Germany`.

`\Placebars` puts one labeled bar into the PICture each time its invoked. In order to run through the entire list of countries you could define `\repeatifnecessary` as follows:

```
13 \def\repeatifnecessary{%
14   \futurelet\next\Repeatifnecessary}
15 \def\Repeatifnecessary{%
16   \ifx /\next
17     \expandafter \finish
18   \else
19     \expandafter \Placebars
20   \fi}
21 \def\finish / {}
```

This is a little complicated, so let's take it one step at a time. The `\futurelet` on line 14 tells TeX to set `\next` equal to the character following whatever country-year-rate entry has just been processed, and to go on to expand `\Repeatifnecessary`. If `\next` is not '/', the `\ifx` on line 16 directs TeX to the "false" clause on line 19. The `\expandafter` there makes TeX expand the `\fi` on line 20 thereby finishing off the `\ifx` ... `\fi` structure. TeX then goes back to the `\Placebars` on line 19, which is now sitting directly in front of the next country-year-rate entry.

On the other hand, if `\next` is '/' (so the entry just processed was 'Switzerland 5 1.5'), the `\ifx` directs TeX to the "true" clause on line 17. The `\expandafter` there leads to the `\else` on line 18; `\else` ... `\fi` expands to ⟨null⟩, because the condition is true. TeX then goes back to the `\finish` on line 17, which gobbles up the '/␣' following the 'Switzerland' entry, thereby preventing these characters from showing up in your PICture. Finally TeX goes on to read whatever comes next, which in this case would be `\endpicture`.

The `\placebars` macro would be especially handy if you were going to make a lot of bar graphs like Figure 6. You should put the definitions of `\placebars`, ..., `\finish` outside a PICture, so that these macros won't vanish when the PICture ends.

What would happen if you were to inadvertently enter '`\placebars /`'? Things would get all screwed up, because the terminator '/' would become the first argument to `\Placebars`. To correct this flaw in the macros, you should change '`\Placebars`' on line 3 to '`\repeatifnecessary`'.

With a view towards generalization, you should make two more simple changes so that the definition of `\Repeatifnecessary` doesn't involve `\Placebars`. Specifically, replace '`\Placebars`' on line 19 by '`\repeatwhat`', and insert

```
\let \repeatwhat=\Placebars
```

between line 2 and (the new) line 3. The repeat structure now can be used with other macros besides `\Placebars`.

To sum up, the solution to the `\placebars` exercise has led to a useful TeXnique for defining upper level plot commands that meet specific needs.

Editor's note: One statement in this article bears repeating: "A large PICture with several curves will exceed the capacity of a standard version of TeX." It happened to this article. TUGboat is produced with a VAX/VMS TeX implementation that has had its memory increased to nearly the maximum possible for a "standard" version: *mem_max* = 65500. To prepare this article successfully, it was necessary to (1) produce each column as a separate page, and put it together using the output driver's electronic pasteup capability; (2) insert explicit page breaks in columns containing especially taxing PICtures; (3) strip unnecessary details out of the header macros (`tugbot.sty`); and (4) run it by itself. Even so, the news with `\tracingstats` turned on was thought-provoking — only 8 words of memory untouched. Michael Wichura is using a C version of TeX without the ordinary memory restrictions.

The LaTeX User's Column

Jackie Damrau
University of New Mexico

I have received four questions since the last column. Please keep those questions or helpful hints coming. Remember, they will be answered as soon as possible via electronic mail (if possible) and then published in the next TUGboat. Until then, happy LaTeXing.

Question 1

Version 2.09 of LaTeX contains the following bug, which I have not seen reported elsewhere, possibly because it would be discovered by a LaTeXnician only when indulging in the aesthetically bizarre practice of using marginal notes and footnotes concurrently. Here it is:

If a `\marginpar` follows a `\footnote` on the same page, then the vertical positioning of the marginal will probably be incorrect. To fix it, the definition of `\@specialoutput` needs to be changed, as shown below, by moving part of the conditional statement `\ifvoid\footins`... so that it is not executed in the case when the output routine is called by `\marginpar`.

```
\def\@specialoutput{%
  \ifnum\outputpenalty > -\@Mii   \@doclearpage
  \else  \ifnum \outputpenalty <-\@Miii
      \ifnum\outputpenalty<-\@MM \deadcycles\z@\fi
      \global\setbox\@holdpg\vbox{\unvbox\@cclv}
    \else \setbox\@tempboxa\box\@cclv
      \@pagedp\dp\@holdpg \@pageht\ht\@holdpg
      \unvbox\@holdpg
% From here:  \ifvoid\footins\else
%      \advance\@pageht\skip\footins
%      \advance\@pagedp\dp\footins
%      \insert\footins{\unvbox\footins}\fi
    \@next\@currbox\@currlist{%
      \ifnum\count\@currbox >\z@
        \ifvoid\footins\else   %  :to here...
          \advance\@pageht\ht\footins
          \advance\@pageht\skip\footins
          \advance\@pagedp\dp\footins
          \insert\footins{\unvbox\footins}\fi %
        \@addtocurcol
      \else \ifvoid\footins\else %  ...and here.
          \insert\footins{\unvbox\footins}\fi %
        \@addmarginpar
      \fi}\@latexbug
    \ifnum\outputpenalty <\z@ \penalty \z@ \fi
  \fi\fi}
```

Chris Rowley
The Open University
Parsifal College
527 Finchley Road
London NW3 7BE

Answer from Leslie Lamport:

Chris Rowley has indeed found a bug in LaTeX, and his fix appears to work. I will probably incorporate it. Relay my thanks to him.

Question 2

Several of us here at the Hughes Aircraft Company Albuquerque Engineering Laboratory are trying to use TeX to produce various sorts of documents. One such application is the making of so-called vugraphs, i.e., landscape orientation charts printed on transparent plastic sheets for use with overhead projectors. The way it works out is that the title or heading of the chart needs to be centered in a field about 4 by 18.5 cm, with text then occupying a separate field about 15 by 25.5 cm. A form has been enclosed to indicate these fields. SLITeX has a lot of good features for this process, but its bias toward portrait mode and vertically centering all text as a unit results in a fair amount of ad hoc and tedious use of `\vspace`, etc. Is there any way to specify separate fields on a slide? Or, better, to describe a subpage by a single command?

We would also be interested in any advice on how to incorporate graphs into the files from which the charts are made.

Richard C. Smith
Hughes Aircraft Company
Albuquerque, New Mexico

Answer from Leslie Lamport:

What Richard Smith wants to do is easy, and there are numerous ways to do it. However, they all involve some understanding of raw TeX. First of all, the vertical centering is achieved by the `\vfil` commands in the definition of `\@makecol` in `slitex.tex`. Removing the `\vfil`'s from this definition will stop SLITeX from vertically centering the slides.

However, there's really no need to remove the vertical centering. Instead, by putting things inside `\vbox`'s of the appropriate height, he can arrange it so that the height of material on a page equals `\textheight`, so vertical centering has no effect. Exactly how he does this depends upon how he wants to enter the input.

The most elegant approach would be to modify the slide environment (by changing the definitions of `\slide` and `\endslide` in `slitex.tex`) so he would just type

`\begin{slide}{}`
The title goes here

```
\midslide
The text field goes here.
\end{slide}
```

This takes a little TEX hacking ability, since one has to begin a `\vbox` in the `\slide` command and end it in the `\midslide`, etc. My guess is that he will need to find a TEX hacker to do this for him, or else take the time to read the TEXbook.

Question 3

I recently sent a query to TEXhax, and Leslie Lamport was good enough to send a reply. Apparently my problems were compounded by my using an out-of-date version of `rep12.sty`. I've subsequently fetched a load of LATEX files from Peter Abbott's software-depository at Aston, and hope that this will help to avoid future problems.

To help people generally avoid such problems, it would be very useful if there was available somewhere a list of the version-numbers/dates of all TEX-related software, so that people could easily check whether they have the most recent version. On the other hand, the range of TEX-related software is now vast, so:

- whoever ran the "somewhere" would have a lot of work to do
- the information would take up a lot of space in TUGboat and/or TEXhax
- difficult questions of where to draw the line might arise. (Are macro package X and style-option Y so popular that the whole world wants to keep their copies up-to-date?)

The problem of where to draw the line is not so difficult for LATEX. Anyone using LATEX will want to keep `lfonts.tex`, `latex.tex`, `lplain.tex` and the standard style- and option-files (the ones mentioned in the book and distributed by Maria Code and Pierre MacKay) up-to-date. Running a "somewhere" that listed the current version-numbers and/or dates for this standard software would I think be manageable.

Would you consider using the **LATEX User's Column** in TUGboat to give the version-numbers/dates of the copies of `lfonts.tex`, `latex.tex`, `lplain.tex`, the standard `*.doc` and the standard `*.sty` that people should be using? I guess that it could be done in about 1.5 column-inches if a suitably small font was used, so it might be acceptable as a standard item in each TUGboat.

Obviously, the problem of keeping non-standard style and options files (e.g. as distributed by Ken Yap) up-to-date is important too, but it seems a less manageable problem and I can't see such an obvious solution.

David Rhead
BITNET:
`David_Rhead@USYSTEM.CCC.NOTTINGHAM.AC.UK`

Question 4

After I sent you my last mail, I sent a copy of the next-to-last paragraph to Leslie Lamport for information. He mailed back a suggestion that the information I need is in `latex.bug`.

So, it would actually be sufficient if people's attention was drawn to the date/version-number of the current `latex.bug`. If they found that they had an out-of-date `latex.bug`, they could take steps to acquire an up-to-date one and, from the up-to-date `latex.bug`, work out what other files they need.

In my situation, for example, I can get files from Aston fairly easily, and Peter Abbott at Aston gets files from `SCORE`, which saves other UK-people having to find them to work out how to get stuff from `SCORE`. I have just fetched `latex.bug_62` from Aston. It mentions stuff up to the end of '87, so it looks fairly up-to-date, but I don't actually know for certain whether or not a more recent `latex.bug` is now available somewhere.

So, would you consider using the **LATEX User's Column** in each TUGboat to state the version-number of the latest `latex.bug`? It need not take more than two sentences, e.g.,

> Details of the current versions and "dates-last-updated" of all the standard LATEX software are to be found in version 63 of `latex.bug`. If you don't already have version 63, acquire it first, so that you can work out what else you need.

David Rhead
BITNET:
`David_Rhead@USYSTEM.CCC.NOTTINGHAM.AC.UK`

Answer (Question 3 and 4)

I have no objections for using my column to announce the above. I would however need someone to supply me with the information. If there is a kind soul out there who would be willing to send me the information, it would be my pleasure to give the LATEX community this added help.

Editor's note: Your editor has access to this information, and will, if reminded, be happy to check the status of `latex.bug` as one of the last chores before putting TUGboat to bed. This is the current status at Score, as of June 30:

```
latex.bug.79  12 May 88  22:44:10
```

Contents of LaTeX Style Collection as of 30th June 1988

Ken Yap
University of Rochester

The LaTeX style collection now contains the files listed below. They are at cs.rochester.edu in directory public/latex-style, and are available for anonymous ftp. You should retrieve the file 00index first to obtain a brief description of current directory contents. The file 00directory contains a reverse time sorted list of files; this may be helpful in keeping your collection in sync with LaTeX-style.

Entries marked with * are new or changed since the last TUGboat listing. Entries marked with a † (ieeetr, acm, siam and apalike BibTeX styles) require BibTeX 0.99b. The others require 0.981 or older.

File	Description
00directory	
00index	
00readme	
a4.sty	Set page size to A4
a4wide.sty	Adjusts width to suit A4
aaai-instructions.tex	Instructions to authors
*aaai-named-0.99.bst	† BiBTeX style to accompany aaai.sty, for version 0.99
*aaai-named-0.98.bst	For version 0.98
aaai.sty	Style file for AAAI conference 1987
acm.bst	† ACM BibTeX style
agugrl.sty	AGU Geophysical Research Letters style, sample
agugrl-sample.tex	
agujgr.sty	AGU Journal of Geophysical Research style, sample
agujgr-sample.tex	
*album.shar	Style for printing cassette labels
alltt.sty	Like verbatim, but permits other commands inside
amssymbols.sty	Load AMS symbol fonts
apalike.doc	American Psychological Association style files
apalike.sty	
*apalike.bst	†
article.txt	Standard files in text format, with places to make language specific changes indicated
art10.txt	
art11.txt	
art12.txt	
biihead.sty	Underlined heading
*bsf.doc	Provide access to bold sans serif fonts in LaTeX
*bsf.sty	
*captcont.sty	Auxiliary file needed by files described in local-suppl
cyrillic.sty	Load cyrillic font
dayofweek.tex	Macros to compute day of week and phase of moon Examples of how to use TeX arithmetic capabilities
deproc.sty	DECUS Proceedings style
deprocldc.tex	Paper that describes the above
docsty.shar	Program to convert .doc to .sty by stripping comments
doublespace.sty	Double spacing in text
draft.sty	Draft option for documents for "debugging"
drafthead.sty	Prints DRAFT in heading
*drop.doc	Style for making large dropped initials for starting paragraphs
*drop.sty	
dvidoc.shar1	Sh archive of DVIDOC, DVI to character device filter for Unix BSD systems
dvidoc.shar2	
epic.shar1	Sh archive of extended picture environment
epic.shar2	
espo.sty	Style file for Esperanto
*fixup.doc	Fixup Plain's \bigl, etc., to track LaTeX size changes
format.sty	Print FP numbers in fixed format
fullpage.doc	Get more out of a page
geophysics.sty	Geophysics journal style
german.sty	Style file for German
ieeetr.bst	† IEEE Transactions BibTeX style
ist21.sty	IST21 document style option for cover page
latex.bug	Latest listing of bugs found in LaTeX
*latex.dif	Differences between versions of standard LaTeX files, beginning 3 Jan 88
layout.readme	Prints nice diagram showing page parameters
layout.tex	
lcustom.tex	Useful macros and definitions for LaTeX
lfonts_ams.readme	Use AMS symbols in LaTeX
lfonts_ams.tex	
lgraph.shar	Sh archive of data to graph command filter in Pascal
local-suppl.tex	Supplement to local guide; describes tgrind, sfwmac, trademark, lcustom, xxxcustom, and xxxslides

*manual.readme	Like "book" but for manuals
*manual.sty	Need to look at "book" for documentation
*man10.sty	
*man11.sty	
*man12.sty	
memo.sty	Memo style option
mfr.sty	Modifier to memo.sty
mitthesis.sty	Massachusetts Institute of Technology thesis format
mitthesis-sample.tex	
natsci.bst	† Natural sciences generic BibTeX style
natsci.sty	Formats citations created with natsci.bst
newalpha.bst	† Modified alphabetic BibTeX style
nl.sty	Style file customized for Dutch
nopagenumbers.doc	Remove page numbers
remark.sty	Like newtheorem but no \it
resume.sty	Format for doing resumes
resume-sample.tex	Sample file
rscsencode.shar	Sh archive of the rscs en/decoder
sc21.sty	ISO/TC97/SC21 document style
sc21-wg1.sty	Option for cover page
*schedule.doc	Style for generating schedule sheets
*schedule.sty	
sfwmac.sty	Useful macros for Unix documentation
*shapiro-btxbst-0.98.doc	
*shapiro-btxbst-0.98.readme	
*shapiro-makebst.sh	A master file for BibTeX styles with standard styles and some new ones. Also a Unix sh script to generate the styles
showlabels.sty	Shows labels and references to them
siam.bib	SIAM BibTeX style
siam.bst	†
siam.doc	SIAM LaTeX style
siam.sty	
siam.tex	
siam10.doc	
siam10.sty	
siam11.sty	
siam12.sty	
slem.doc	Change \sl to \em
spacecites.doc	Modified to give spacing between citations
suthesis.doc	Stanford U thesis style
suthesis.sty	
*svma.sty	Style for Springer-Verlag reports, multi-author
*svma.tex	Manual, Springer-Verlag
*svsa.sty	Springer-Verlag, single author
*tabledoc.tex	Documentation for tables.sty
*tables.sty	Ruled and unruled tables made easy
texindex.shar	Style file and processor for index entries for VMS
texnames.doc	Define a couple more TeX names
texnames.sty	
tgrind.sty	Tgrind macros for LaTeX instead of TeX
threepart.sty	Three part page headers
titlepage.txt	Style file in text format to go with article.txt
trademark.sty	Definitions of common trademarks
uct10.doc	U of California thesis style
uct11.doc	
uct12.doc	
ucthesis.doc	
ucthesis.readme	
*uct10.diffs	Context diffs from LaTeX report style files to make updating easy
*uct11.diffs	
*uct12.diffs	
*ucthesis.diffs	
*uuencod.shar	Sh archive of the uu en/decoder
vdm.doc	Vienna Development Method LaTeX style
vdm.sty	
vdm.tex	
wsltex.shar	Wordstar to LaTeX filter, C and Pascal versions
xxxcustom.tex	Supplementary macros for xxx-tex, for some xxx
xxxslides.sty	Supplementary macros for SliTeX, includes slides.sty

More submissions are very welcome. Send them to

Ken
LaTeX-Style@cs.rochester.edu
..!rochester!latex-style

Editor's note: People sending future submissions should note that some gateways to Bitnet strip off everything beyond 80 columns, and perhaps corrupt some other data as well (ASCII tabs may or may not remain intact). Please structure your file so that it will survive.

For Internet users: how to ftp

An example session is shown below. Disclaimer: ftp syntax varies from host to host. Your syntax may be different. The syntax presented here is that of Unix ftp. Comments in parentheses.

Non-Internet users: how to retrieve by mail

An archive server for LaTeX files has been installed. Send a piece of mail to `LaTeX-Style` (`@cs.rochester.edu`, via UUCP or your favourite gateway) in the following format.

- Subject line should contain the phrase "`@file request`".
- The body of the mail should start with a line containing only an `@` (at) sign.

Important! The first line following the "at" line should be a mail address **from** Rochester **to** you. (Undeliverable mail will be silently dropped on the floor.)

- Follow your return address by the names of the files you want, either one to each line, or many to each line, separated by spaces.
- End with a line containing only an `@` sign.
- Case is not significant.

For example, if you are `user` at `site.bitnet`, this is what you should send: (*don't forget your address!*)

```
To: latex-style@cs.rochester.edu
Subject: @file request

@
user%site.bitnet@cunyvm.cuny.edu
00readme
00index
@
```

A word to the wise: it is best to fully qualify your mail address. Our mailer is pretty ignorant of Bitnet, CSnet or UUCP addresses unless they are in registered domains. It is best that you supply explicit gateway routes. Use the new domainized form of addresses whenever possible. Examples:

```
user%site.bitnet@cunyvm.cuny.edu
user%site.csnet@relay.cs.net
site!user@uunet.uu.net
```

Long UUCP paths are discouraged. System administrators get upset and your turnaround is very slow anyway.

If the `Subject:` line looks like:

```
Subject: @file request uuencode
```

or

```
Subject: @file request rscsencode
```

then the mail will be encoded with the requested scheme before sending. This *might* help sites that get mail through gateways with unfriendly EBCDIC/ASCII mappings. You can find sources for the two types of en/decoders in the collection. You may have to do some porting of sources.

Be patient as the server is actually a batch program run once a day. Files will be sent in batches, each not exceeding 100kbytes in size.

Distribution for IBM PC and clone users

There are two sources.

- David W. Hopper
  446 Main Street
  Toronto, Ontario
  Canada M4C 4Y2

has LaTeX style files only.

1. Either one 1.2 MB diskette or three 360 KB diskettes, blank and formatted.
2. Indication of the format required,
3. A self-addressed mailer, and
4. A $5.00 donation per set of files, to cover postage and equipment wear & tear. (If you live outside North America, airmail delivery will probably require more postage. You should probably contact David for details.)

Sample FTP session for Internet users

```
% ftp cayuga.cs.rochester.edu   (a.k.a. cs.rochester.edu, a.k.a. 192.5.53.209)
  ...                           (general blurb)
user: anonymous
password: <any non-null string>
ftp> cd public/latex-style      (where the files are)
ftp> ls                         (to see what is there)
  ...                           (lots of output)
ftp> get 00index
  ...                           (more blurb)
ftp> quit
```

5. No phone calls or personal visits please.

- Jon Radel
  P. O. Box 2276
  Reston, VA 22090

has LaTeX style files and other goodies. For a list or other info send a SASE.

1. 360 KB diskettes, blank and formatted.
2. A stamped, self-addressed mailer, and
3. $1.50 per disk. If you live outside North America, skip the stamps and send additional money or International Reply Coupons.

As a convenience for people who have more money than floppies, Jon will supply everything for $6.00 per disk to U.S./Canada/Mexico addresses.

Editor's note: Traffic on the network servers and gateways has been very high recently, and in order to provide improved service, there have been some volunteers to maintain local "slave" repositories of the LaTeX style collection. There is usually a geographic or network restriction requested, since the idea is to cut down traffic, not add to it. The following areas will be covered by the volunteers listed.

- Bitnet users: Texas A&M maintains a list- and file-server which is already handling (with TEX-L) much of the Bitnet distribution of TeXhax. An inquiry via listserv will retrieve a list of all TeX-related files:

```
tell listserv at tamvm1 get tex filelist
```

Additional volunteers should contact Ken.

A Note on Processing Parts With LaTeX

Stephan v. Bechtolsheim

In this note I would like to suggest how to administer efficiently the processing of a multi-part LaTeX-based document. I have assumed that the document is fairly large and therefore LaTeX's `\includeonly` feature is used — this note does **not** apply to documents which consist of only one LaTeX file.

Two implementations of my ideas will be shown, the first using UNIX and the other using MS/DOS. To derive an implementation to run under yet another operating system should be straightforward.

Let us assume that `part1.tex`, `part2.tex` and `part3.tex` are the three parts of a document. There is also a *main source file* called `main.tex`, which might look as follows:

```
\documentstyle{article}
\includeonly{part1}
\begin{document}
    \include {part1}
    \include {part2}
    \include {part3}
\end{document}
```

Assume further that the user always processes only one part at a time, never two or more, or the complete document in one piece. Then it is natural to rename `main.dvi` after running LaTeX to `part1.dvi`, `part2.dvi` or `part3.dvi`, depending on which part of the document was processed. There is, as far as the user is concerned, no `main.dvi` any more. `main.log` is also renamed, and becomes either `part1.log`, `part2.log` or `part3.log`.

Renaming the parts has an additional advantage: it is easy to find out whether or not a part still has to be processed by LaTeX. If `part1.tex` was last changed at `2:00pm` and the time stamp of `part1.dvi` shows `3:00pm` then `part1.dvi` is "up-to-date". But if `part1.dvi`'s write time is `1:00pm` then `part1.tex` has to be reprocessed because the source file is newer than its `dvi` file.

The next step is to rewrite the original main source file as follows (the `\includeonly` statement is now read in from an external file `include.tmp`):

```
\documentstyle{article}
\input{include.tmp}    % CHANGED
\begin{document}
    \include {part1}
    \include {part2}
    \include {part3}
\end{document}
```

A little UNIX *shell script* (the UNIX term for a command procedure) palatex might look as follows (the shell script has one argument $1 which is the part of the document to be processed, without the file extension tex):

```
# UNIX palatex shell script
echo "\includeonly{$1}" > include.tmp
latex main
mv main.dvi $1.dvi
mv main.log $1.log
```

The above shell script would be, for instance, invoked as palatex part1. Two remarks with respect to this script at this point:

1. The name for the main source file in the shell script is fixed to main.tex. I normally store every document in a separate directory and I find it very convenient always to call the main source file by the same name. Naturally, the shell script could be modified to accommodate another parameter, which is the name of the main source file.
2. This idea can be applied even more conveniently in a makefile. Now all you have to type is make, and only those parts of the document which have to be processed will be processed. Here is such a makefile:

```
.SUFFIXES:    .dvi    .tex
FILES = part1.dvi part2.dvi\
                  part3.dvi

all:    $(FILES)

.tex.dvi:
    echo '\includeonly{$*}'>\
              include.tmp
    latex main
    mv main.dvi $*.dvi
    mv Main.log $*.log
```

Now let us write a batch file palatex.bat for MS/DOS along the same lines. In MS/DOS, %1 stands for the first parameter in a batch procedure.

```
if exists %1.dvi del %1.dvi
if exists %1.log del %1.log
echo \includeonly{%1} > include.tmp
tex &lplain main
ren main.dvi %1.dvi
ren main.log %1.log
```

Observe that when using palatex.bat you type the name of the part *without* the file extension tex; so you type palatex.bat part1 and **not** palatex part1.tex. If you type the file name with the file extension tex then you will have a catastrophe. Assume you typed palatex part1.tex. Now the first line of the procedure expands to:

```
if exists part1.tex.dvi
                  del part1.tex.dvi
```

(Observe the illegal file names formed this way.) MS/DOS will interpret this as:

```
if exists part1.tex
                  del part1.tex
```

In other words, your source part1.tex is removed!

I am not an MS/DOS person but I encourage knowledgeable MS/DOS people to write a little C-program which aborts the command procedure if the name of a part was entered with a file extension. By the way, there is also a make program available for MS/DOS. Using that, you can automate the whole procedure under MS/DOS the same way I outlined it for UNIX.

Sue Volkmann from the *Woods Hole Oceanographic Institute* in Woods Hole, MA, helped me implement the ideas under MS/DOS.

Page Layout in LaTeX: Erratum

Kent McPherson
SLI Avionic Systems Corp.

In LAYOUT.STY, which was printed in Volume 9, No. 1, there were a couple of commented lines. Unfortunately, when the style file was printed in two-column mode, these commented lines were broken into two lines each, and the second line was then uncommented. This causes the macros to fail.

The lines in question appear in the middle of column 2, page 81. They look like:

```
%\omarginref=\omargin
  \advance\omarginref by \oneinch
%\advance\omarginref by \hofref
```

and

```
%\emarginref=\emargin
  \advance\emarginref by \oneinch
%\advance\emarginref by \hofset
```

These lines should be deleted from LAYOUT.STY.

Editor's note: We are sorry for any inconvenience that resulted from our omission of the comment character from the wrapped lines shown above, and wish to thank those of you who reported problems.

Calendar

1988

Aug	1 – 5	ACM SIGGRAPH; Atlanta, Ga. For information, call (312) 644-6610

California State College, Northridge

Aug	1 – 5	Intensive Beginning/Intermed. TeX
Aug	8 – 12	Advanced TeX/Macro Writing

University of New Mexico, Albuquerque

Aug	1 – 5	Beginning TeX
Aug	1 – 5	Intensive Beginning/Intermed. TeX
Aug	8 – 12	Advanced TeX/Macro Writing

Rutgers University, Busch Campus, Piscataway, New Jersey

Aug	1 – 5	Intensive LaTeX
Aug	8 – 12	Advanced TeX/Macro Writing
Aug	15 – 19	Beginning TeX
Aug	15 – 19	Intensive Beginning/Intermed. TeX

TeX Users Group 1988 Conference McGill University, Montréal, Québec

Aug	15 – 19	Beginning TeX
Aug	15 – 19	Intensive Beginning/Intermed. TeX
Aug	16 – 19	Short Course in METAFONT
Aug	18 – 19	Short Course in PostScript
Aug	22 – 24	**TUG Annual Meeting** See announcement, page 000.
Aug	25 – 26	Short Course: Macro Writing
Aug	25 – 26	Short Course: Output Routines

Stanford University, Stanford, California

Aug	29 – Sep 2	Beginning TeX
Aug	29 – Sep 2	Intensive Beginning/Intermed. TeX
Sep	6 – 10	Advanced TeX/Macro Writing

Aug	22	DECUS Regional Seminar, Tysons Corner, Virginia. TeX Made Easy — Using TeX with the Plain macro package, Dan Zirin. For information, contact Diane Spirio, (617) 480-3307.
Sep	12	**TUGboat Volume 9, No. 3:** Deadline for receipt of manuscripts.

University of Exeter, England

Sep	5 – 9	Intensive Beginning/Intermed. TeX
Sep	19 – 23	Advanced TeX/Macro Writing For information, contact Clive Nicholas, (0392) 411907.

University of Illinois, Chicago, Illinois

Sep	12 – 16	Beginning TeX
Sep	12 – 16	Intensive Beginning/Intermed. TeX
Sep	12 – 16	Advanced TeX/Macro Writing

Oct	2 – 6	Electronic Design in Print '88 Conference. San Diego Convention Center, San Diego, California. For information, contact Electronic Design in Print '88, Vista, CA, (619) 758-9460.
Oct	17 – 21	1988 Electronic Printer and Publishing Conference. Fairmont Hotel, San Jose, California. For information, contact CAP International, Marshfield, MA, (617) 837-1341.
Dec	5 – 9	ACM Conference on Document Processing Systems, Santa Fe, New Mexico. For information, contact Peter Orbeton, (617) 577-8500 or `Orbeton.chi@xerox.com`.

For additional information on the events listed above, contact the TUG office (401-751-7760) unless otherwise noted.

Status as of 1 July 1988

GUTenberg Meeting Report

Bernard Gaulle
(EARN: UCIR001 at FRORS31)

A meeting of the Group of French Users of TeX (named of course **GUTenberg** — Groupe francophone des Utilisateurs de TeX) was held in Paris April 26, at IRESCO.*

First, let me recall a few historical facts:

GUTenberg, for those of you who are not familiar with it, was born in 1984. His christening was a "secret ceremony" attended by less than ten persons. In spite of his very young age, his fame grew rapidly among scientific research communities: he graduated in humanities last year. A great feast was organized in his honor with almost 160 guests. He obtained a Master of Arts in Typography a few months ago and agreed to give his name to a group of publishers and scientific authors. This talented child is now very anxious to get married ... !

But let us turn to more serious matters:

When I decided, early in February, to organize a meeting for TeX users, I wrote to Ray Goucher asking for TUG support. My request was transmitted to the TUG executive board who unanimously agreed 'in principle', but felt that a request for support had to be submitted at least 6 months in advance! I answered that, although such a delay may consolidate the position of well established organizations, it would hardly encourage small, informal, and recently created groups (like **GUTenberg**) to expand and acquire financing.

In spite of this, THE meeting was held in April as originally announced. I wish to express my acknowledgements to all the people who contributed to the success of this little congress, and particularly to J. André, P.O. Flavigny, L. Carnes, D. Collin, and their respective companies/organizations: IRISA, IRESCO-LASMAS, Personal TeX Inc., and TRÉMA. The congress room was full with more than 120 invited persons.

Quite a number of lectures were on the agenda and some time was to be devoted to further debate, especially concerning **GUTenberg**'s future.

I opened the meeting by explaining the goals which had been set for the day: exchange experiences, inquire about the future of TeX, and find volunteers who agree to dedicate a little of their time to **GUTenberg**.

The day started with a talk by Michael Ferguson about multilingual MLTeX and INRSTeX used in a bilingual environment. Other people confirmed later on that MLTeX is a very interesting tool for typesetting French. (Personally, I wish to add that we do highly appreciate the multilingual topics in TUGboat.**)

André Lannes, a consultant in a new company he helped to create, explained how TeX can be used with conviviality on a PC when one assembles a good text editor with many programmed keys, a little syntax preprocessor and a good previewer. At this point, F. Chahuneau expressed his own opinion, which is that displaying pre-formatted macros at the screen with some fields to be filled in should not be described as "friendly" to the many publishers in the audience who may just have discovered TeX.

Very interesting lectures were delivered by Nicolas Brouard, Daniel Taupin, and Philippe Louarn on how to publish specialized magazine, book, or annual scientific reports. Specific difficulties were dealt with by Daniel Taupin: diverse overflow problems of TeX and/or driver implementations on PC and CIRCÉ*** mainframes.

Marion Seltz-Laurière (who has created a "TeX for beginners" course together with Denis Corroyer) spoke about her successful experience. She added that she lacked support to organize another course.

The afternoon started with speeches about Mac and PC related software. Laurent Siebenmann, developer of Sweet-TeX, made a comparison between TeXtures and MacTeX and added a few words on preprocessors. Lance Carnes made an inventory of all products distributed by Personal TeX Inc. with their different versions and release status.

Professional publishers then explained their opinion about TeX. Daniel Collin, who two years ago created the first French composition company based on TeX (which also distributes Personal TeX Inc. products), showed us many outstandingly beautiful examples of his production and explained — with great modesty — how they had been created and input. His company works mostly for publishers and has acquired much experience in the fields of education and science.

François Chahuneau, who recently joined a large publisher (Berger Levrault) explained very precisely the pros and cons of TeX. He said why he thinks TeX barely has any future at all among private business organizations: as it belongs to the public domain, no famous company may support and maintain TeX officially.

* Institut de REcherche du CNRS sur les Sociétés COntemporaines

** Editor's note: We continue to invite articles on such subjects from users of TeX in all languages.

*** National computing center of CNRS.

Members of the first French TEX printing company (Louis-Jean) attended the meeting and M. Laugier gave us the reasons why he uses TEX: reduced cost, typographic quality, and ability to produce new fonts with METAFONT. He insisted on the fact that TEX should be used exclusively for what it was designed: mathematical and literary books. Other styles he called "fancies" should be produced by other technical and traditional means.

Victor Ostromoukhov, who has implemented METAFONT and other tools on Mac and contributed to the start of TEX printing at Louis-Jean, told us about his developments. A discussion took place about printing resolutions. Three ideas were expressed: according to one member of the audience, 1200dpi is not necessary, another said he was able to distinguish two dots at 600, and the last explained that laser printing can not be fully satisfactory at any resolution, laser impact not being clean enough.

A great deal of the time supposedly devoted to the debate on **GUT**enberg's future had unfortunately already fled by and we only decided that all people who are interested in regular meetings, willing to contribute to number 1 of "Cahiers **GUT**enberg" (number 0 was handed out at the congress) or eager to exchange macros and TEXniques would get in touch via the above electronic mail address.

The congress ended with a cocktail offered by Personal TEX, Inc.

After all the encouraging contacts I had during the meeting, I feel entitled to say that **GUT**enberg is growing steadily and should soon become an "adult".

Late-Breaking News

Knuth Scholarship Winner

The third annual Knuth Scholarship has been awarded to Larry Sharlow, a Technical Writer at Orban Associates Inc. of San Francisco. Orban Associates designs and manufactures audio processing equipment for broadcast, recording, and sound reinforcement. This year's scholarship is sponsored by ArborText, Ann Arbor, Michigan, and will enable Larry to attend the Annual Meeting in Montreal and a two-day TEX short course.

Sharlow's responsibilities include researching, editing, and writing technical manuals, bulletins, and instruction sheets. In addition to implementing the software used to produce these materials, he coordinates their review and production. Over a period of a year and a half, he developed his own macropackage, ORBTEX, to facilitate typesetting at Orban Associates.

The judges were impressed with the simplicity, appropriateness, and effectiveness of the macros, as well as the completeness of documentation. Special attention was taken to place these macros in context not only with TEX but with the total working environment, which included various hardware and software. Larry's work is a good example of the independent mastery of TEX. This is one important function of the TEX Users Group — to foster the communication between people working independently with TEX.

Institutional Members

Addison-Wesley Publishing Company, *Reading, Massachusetts*

The Aerospace Corporation, *El Segundo, California*

American Mathematical Society, *Providence, Rhode Island*

ArborText, Inc., *Ann Arbor, Michigan*

ASCII Corporation, *Tokyo, Japan*

Aston University, *Birmingham, England*

Brookhaven National Laboratory, *Upton, New York*

Brown University, *Providence, Rhode Island*

California Institute of Technology, *Pasadena, California*

Calvin College, *Grand Rapids, Michigan*

Centre Inter-Régional de Calcul Électronique, CNRS, *Orsay, France*

City University of New York, *New York, New York*

College of St. Thomas, Computing Center, *St. Paul, Minnesota*

College of William & Mary, Department of Computer Science, *Williamsburg, Virginia*

Columbia University, Center for Computing Activities, *New York, New York*

COS Information, *Montreal, P. Q., Canada*

Data General Corporation, *Westboro, Massachusetts*

DECUS, L&T Special Interest Group, *Marlboro, Massachusetts*

Department of National Defence, *Ottawa, Ontario, Canada*

Digital Equipment Corporation, *Nashua, New Hampshire*

dit Company, Ltd., *Tokyo, Japan*

Edinboro University of Pennsylvania, *Edinboro, Pennsylvania*

Electricité de France, *Clamart, France*

Environmental Research Institute of Michigan, *Ann Arbor, Michigan*

European Southern Observatory, *Garching bei München, Federal Republic of Germany*

Fermi National Accelerator Laboratory, *Batavia, Illinois*

Ford Aerospace & Communications Corporation, *Palo Alto, California*

Försvarets Materielverk, *Stockholm, Sweden*

General Motors Research Laboratories, *Warren, Michigan*

Geophysical Company of Norway A/S, *Stavanger, Norway*

Grinnell College, Computer Services, *Grinnell, Iowa*

Grumman Corporation, *Bethpage, New York*

GTE Laboratories, *Waltham, Massachusetts*

Hart Information Systems, *Austin, Texas*

Hartford Graduate Center, *Hartford, Connecticut*

Harvard University, Computer Services, *Cambridge, Massachusetts*

Hewlett-Packard Co., *Boise, Idaho*

Hobart & William Smith Colleges, *Geneva, New York*

Humboldt State University, *Arcata, California*

Hutchinson Community College, *Hutchinson, Kansas*

IBM Corporation, Scientific Center, *Palo Alto, California*

Illinois Institute of Technology, *Chicago, Illinois*

Imagen, *Santa Clara, California*

Institute for Advanced Study, *Princeton, New Jersey*

Institute for Defense Analyses, Communications Research Division, *Princeton, New Jersey*

Intergraph Corporation, *Huntsville, Alabama*

Intevep S. A., *Caracas, Venezuela*

Iowa State University, *Ames, Iowa*

Istituto di Cibernetica, Università degli Studi, *Milan, Italy*

Kuwait Institute for Scientific Research, *Safat, Kuwait*

Los Alamos National Laboratory, University of California, *Los Alamos, New Mexico*

Louisiana State University, *Baton Rouge, Louisiana*

Marquette University, Department of Mathematics, Statistics, and Computer Science, *Milwaukee, Wisconsin*

Massachusetts Institute of Technology, Artificial Intelligence Laboratory, *Cambridge, Massachusetts*

Massachusetts Institute of Technology, Information Services, *Cambridge, Massachusetts*

Mathematical Reviews, American Mathematical Society, *Ann Arbor, Michigan*

Max Planck Institute Stuttgart, *Stuttgart, Federal Republic of Germany*

McGill University, *Montreal, Quebec, Canada*

National Cancer Institute, *Frederick, Maryland*

National Center for Atmospheric Research, *Boulder, Colorado*

National Institutes of Health, *Bethesda, Maryland*

National Research Council Canada, Computation Centre, *Ottawa, Ontario, Canada*

New Jersey Institute of Technology, *Newark, New Jersey*

New York University, Academic Computing Facility, *New York, New York*

Northeastern University, Academic Computing Services, *Boston, Massachusetts*

Online Computer Library Center, Inc. (OCLC), *Dublin, Ohio*

Pennsylvania State University, Computation Center, *University Park, Pennsylvania*

Personal TEX, Incorporated, *Mill Valley, California*

Purdue University, *West Lafayette, Indiana*

QMS, Inc, *Mobile, Alabama*

Queens College, *Flushing, New York*

Research Triangle Institute, *Research Triangle Park, North Carolina*

RE/SPEC, Inc., *Rapid City, South Dakota*

Rice University, *Department of Computer Science, Houston, Texas*

Ruhr Universität Bochum, *Bochum, Federal Republic of Germany*

Rutgers University, Hill Center, *Piscataway, New Jersey*

St. Albans School, *Mount St. Alban, Washington, D.C.*

Sandia National Laboratories, *Albuquerque, New Mexico*

SAS Institute, *Cary, North Carolina*

Schlumberger Well Services, *Houston, Texas*

Science Applications International Corp., *Oak Ridge, Tennessee*

I. P. Sharp Associates, *Palo Alto, California*

Smithsonian Astrophysical Observatory, Computation Facility, *Cambridge, Massachusetts*

Software Research Associates, *Tokyo, Japan*

Sony Corporation, *Atsugi, Japan*

Space Telescope Science Institute, *Baltimore, Maryland*

Springer-Verlag, *Heidelberg, Federal Republic of Germany*

Stanford Linear Accelerator Center (SLAC), *Stanford, California*

Stanford University, Computer Science Department, *Stanford, California*

Stanford University, ITS Graphics & Computer Systems, *Stanford, California*

State University of New York, Department of Computer Science, *Stony Brook, New York*

Stratus Computer, Inc., *Marlboro, Massachusetts*

Syracuse University, *Syracuse, New York*

Talaris Systems, Inc., *San Diego, California*

Texas A & M University, Computing Services Center, *College Station, Texas*

Texas A & M University, Department of Computer Science, *College Station, Texas*

Tribune TV Log, *Glens Falls, New York*

TRW, Inc., *Redondo Beach, California*

Tufts University, *Medford, Massachusetts*

TV Guide, *Radnor, Pennsylvania*

TYX Corporation, *Reston, Virginia*

UNI.C, Danmarks EDB-Center, *Aarhus, Denmark*

University College, *Cork, Ireland*

University of Alabama, *Tuscaloosa, Alabama*

University of British Columbia, Computing Centre, *Vancouver, British Columbia, Canada*

University of British Columbia, Mathematics Department, *Vancouver, British Columbia, Canada*

University of Calgary, *Calgary, Alberta, Canada*

University of California, Berkeley, Academic Computing Services, *Berkeley, California*

University of California, Berkeley, Computer Science Division, *Berkeley, California*

University of California, Irvine, Department of Mathematics, *Irvine, California*

University of California, Irvine, Information & Computer Science, *Irvine, California*

University of California, San Diego, *La Jolla, California*

University of California, San Francisco, *San Francisco, California*

University of Chicago, Computation Center, *Chicago, Illinois*

University of Chicago, Computer Science Department, *Chicago, Illinois*

University of Chicago, Graduate School of Business, *Chicago, Illinois*

University of Crete, Institute of Computer Science, Research Center, *Heraklio, Crete, Greece*

University of Delaware, *Newark, Delaware*

University of Exeter, Computer Unit, *Exeter, Devon, England*

University of Glasgow, *Glasgow, Scotland*

University of Groningen, *Groningen, The Netherlands*

University of Illinois at Chicago, Computer Center, *Chicago, Illinois*

University of Illinois at Urbana-Champaign, Computer Science Department, *Urbana, Illinois*

University of Kansas, Academic Computing Services, *Lawrence, Kansas*

University of Maryland, *College Park, Maryland*

University of Massachusetts, *Amherst, Massachusetts*

University of North Carolina, School of Public Health, *Chapel Hill, North Carolina*

University of Oslo, Institute of Informatics, *Blindern, Oslo, Norway*

University of Ottawa, *Ottawa, Ontario, Canada*

University of Southern California, Information Sciences Institute, *Marina del Rey, California*

University of Stockholm, Department of Mathematics, *Stockholm, Sweden*

University of Tennessee at Knoxville, Department of Electrical Engineering, *Knoxville, Tennessee*

University of Texas at Austin, Physics Department, *Austin, Texas*

University of Texas at Dallas, Center for Space Science, *Dallas, Texas*

University of Vermont, *Burlington, Vermont*

University of Washington, Department of Computer Science, *Seattle, Washington*

University of Western Australia, Regional Computing Centre, *Nedlands, Australia*

University of Wisconsin, Academic Computing Center, *Madison, Wisconsin*

Uppsala University, *Uppsala, Sweden*

Vanderbilt University, *Nashville, Tennessee*

Vereinigte Aluminium-Werke AG, *Bonn, Federal Republic of Germany*

Villanova University, *Villanova, Pennsylvania*

Vrije Universiteit, *Amsterdam, The Netherlands*

Washington State University, *Pullman, Washington*

Widener University, Computing Services, *Chester, Pennsylvania*

John Wiley & Sons, Incorporated, *New York, New York*

Worcester Polytechnic Institute, *Worcester, Massachusetts*

Yale University, Computer Center, *New Haven, Connecticut*

Yale University, Department of Computer Science, *New Haven, Connecticut*

TeX Users Group **1988** **Membership Form**

Request for Information

The TeX Users Group maintains a database and publishes a membership list containing information about the equipment on which members' organizations plan to or have installed TeX, and about the applications for which TeX would be used. This list is updated periodically and distributed to members with TUGboat, to permit them to identify others with similar interests. Thus, it is important that the information be complete and up-to-date.

Please answer the questions below, in particular those regarding the status of TeX and the hardware on which it runs or is being installed. (Operating system information is particularly important in the case of IBM mainframes and VAX.) This hardware information is used to group members in the listings by computer and output device.

If accurate information has already been provided by another TUG member at your site, you may indicate that member's name, and the information will be repeated.

If your current listing is correct, you need not answer these questions again. Your cooperation is appreciated.

- *Send completed form with remittance*
  (checks, money orders, UNESCO coupons) to:
  TeX Users Group
  P. O. Box 594
  Providence, Rhode Island 02901, U.S.A.
- *For foreign bank transfers*
  direct payment to the TeX Users Group,
  account #002-031375, at:
  Rhode Island Hospital Trust National Bank
  One Hospital Trust Plaza
  Providence, Rhode Island 02903-2449, U.S.A.
- *General correspondence*
  about TUG should be addressed to:
  TeX Users Group
  P. O. Box 9506
  Providence, Rhode Island 02940-9506, U.S.A.

Name: ____________________

Home [ ]
Bus. [ ] Address: ____________________

____

____

____

QTY	ITEM	AMOUNT
	1988 TUGboat Subscription/TUG Membership (Jan.–Dec.) – **North America** New (first-time): [ ] $30.00 each Renewal: [ ] $40.00; [ ] $30.00 – reduced rate if renewed before February 1, 1988	
	1988 TUGboat Subscription/TUG Membership (Jan.–Dec.) – **Outside North America** New (first-time): [ ] $40.00 each Renewal: [ ] $45.00; [ ] $40.00 – reduced rate if renewed before February 1, 1988	
	TUGboat back issues, 1980 (v. 1) #1; 1981 (v. 2) #1,2,3; 1982 (v. 3) #1,2; 1983 (v. 4) #1,2; 1984 (v. 5) #1,2; 1985 (v. 6) #1,2,3; 1986 (v. 7) #1,2,3; 1987 (v. 8) #1,2,3 $15.00 per issue (19), circle issue(s) desired:	

Air mail postage is included in the rates for all subscriptions and memberships outside North America.
Quantity discounts available on request.

TOTAL ENCLOSED: ____________
(Prepayment in U.S. dollars required)

* * * *

Membership List Information

Institution (if not part of address):

Title:
Phone:
Network address: [ ] Arpanet [ ] BITnet
[ ] CSnet [ ] uucp

Specific applications or reason for interest in TeX:

My installation can offer the following software or technical support to TUG:

Please list high-level TeX users at your site who would not mind being contacted for information; give name, address, and telephone.

Date:
Status of TeX: [ ] Under consideration
[ ] Being installed
[ ] Up and running since
Approximate number of users:
Version of TeX: [ ] SAIL
Pascal: [ ] TeX82 [ ] TeX80
[ ] Other (describe)

From whom obtained:

Hardware on which TeX is to be used:

Computer(s)	Operating system(s)	Output device(s)

Please answer the following questions regarding output devices used with TEX
if this form has never been filled out for your site, or if you have new information.
Use a separate form for each output device.

Name ________________________ Institution ________________________

A. Output device information
Device name
Model
1. Knowledgeable contact at your site
Name
Telephone
2. Device resolution (dots/inch)
3. Print speed (average feet/minute in graphics mode)
4. Physical size of device (height, width, depth)

5. Purchase price
6. Device type
[ ] photographic [ ] electrostatic
[ ] impact [ ] other (describe)

7. Paper feed [ ] tractor feed
[ ] friction, continuous form
[ ] friction, sheet feed [ ] other (describe)

8. Paper characteristics
a. Paper type required by device
[ ] plain [ ] electrostatic
[ ] photographic [ ] other (describe)

b. Special forms that can be used [ ] none
[ ] preprinted one-part [ ] multi-part
[ ] card stock [ ] other (describe)

c. Paper dimensions (width, length)
maximum
usable
9. Print mode
[ ] Character: ( ) Ascii ( ) Other
[ ] Graphics [ ] Both char/graphics
10. Reliability of device
[ ] Good [ ] Fair [ ] Poor
11. Maintenance required
[ ] Heavy [ ] Medium [ ] Light
12. Recommended usage level
[ ] Heavy [ ] Medium [ ] Light
13. Manufacturer information
a. Manufacturer name
Contact person
Address

Telephone
b. Delivery time
c. Service [ ] Reliable [ ] Unreliable

B. Computer to which this device is interfaced
1. Computer name
2. Model
3. Type of architecture *
4. Operating system

C. Output device driver software
[ ] Obtained from Stanford
[ ] Written in-house
[ ] Other (explain)

D. Separate interface hardware (if any) between host computer and output device (e.g. Z80)
1. Separate interface hardware not needed because:
[ ] Output device is run off-line
[ ] O/D contains user-programmable micro
[ ] Decided to drive O/D direct from host
2. Name of interface device (if more than one, specify for each)

3. Manufacturer information
a. Manufacturer name
Contact person
Address

Telephone
b. Delivery time
c. Purchase price
4. Modifications
[ ] Specified by Stanford
[ ] Designed/built in-house
[ ] Other (explain)

5. Software for interface device
[ ] Obtained from Stanford
[ ] Written in-house
[ ] Other (explain)

E. Fonts being used
[ ] Computer Modern
[ ] Fonts supplied by manufacturer
[ ] Other (explain)

1. From whom were fonts obtained?

2. Are you using Metafont? [ ] Yes [ ] No

F. What are the strong points of your output device?

G. What are its drawbacks and how have you dealt with them?

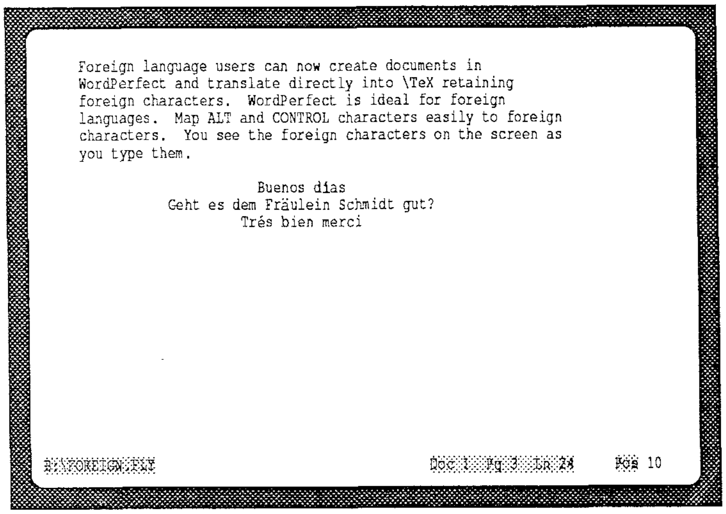
H. Comments – overview of output device

*If your computer is "software compatible" with another type (e.g. Amdahl with IBM 370), indicate the type here.

Publishing Companion 1.03

Desktop Publishing Has Never Been Simpler and Will Never Be the Same

Foreign Language Support



```
Foreign language users can now create documents in
WordPerfect and translate directly into \TeX retaining
foreign characters.  WordPerfect is ideal for foreign
languages.  Map ALT and CONTROL characters easily to foreign
characters.  You see the foreign characters on the screen as
you type them.

                    Buenos días
        Geht es dem Fräulein Schmidt gut?
                  Trés bien merci











B:\FOREIGN.FLY                              Doc 1 Pg 3 Ln 24    Pos 10
```

What you see on the Word-Perfect screen

Is fully translated to TEX as shown below

Foreign language users can now create documents in WordPerfect and translate directly into TEX retaining foreign characters. WordPerfect is ideal for foreign languages. Map ALT and CONTROL characters easily to foreign characters. You see the foreign characters on the screen as you type them.

Buenos días
Geht es dem Fräulein Schmidt gut?
Trés bien merci

K·TALK®
COMMUNICATIONS

3920 Olentangy River Road
Columbus, Ohio 43214 U.S.A.
(614) 459-9711

European dealers:
Bruce Wolman of KAOS A/S, Boks 3169 Elisenberg, 0208 Oslo 2 NORWAY -- Telephone +47 02-59-02-94
Knud Lønsted of Interbase, Dantes Plads 1, DK-1556 København V Denmark -- Telephone +45 01-93-28-29
Ewart North of UniTEX Systems, 12 Dale View Road, Beauchief, Sheffield S8 0EJ. -- Telephone (0742) 351 489

Publishing Companion priced at $179.00. Screen capture program used for above screen available for $79.00. Add shipping and handling of $4.00 for US & Canada or $35.00 for Europe. Prepayment or purchase order required. A remittance from outside the United States must be payable in U.S. dollars and can be an international money order or a check drawn on a U.S., Canadian or European bank.

Printed in U.S.A. 8802-1

Quotations from *Micro/Systems Journal*, June 1988:

The Scientific Computer User
Recent News About TEX

. . .

"MacroTEX. . . is intended for general use and thus there are a number of different modules intended for different purposes. The package has been set up so that each module is read into TEX's memory only when it is needed, so if you only want to use a small number of the modules, you will suffer relatively little interference with the use of other large macro packages at the same time.

". . . What is striking about MacroTEX is that the definitions of Plain TEX are almost entirely untouched, so you do not lose flexibility.

". . . Like LaTEX, MacroTEX is concerned with the preparation of manuscripts in different styles. The styles supplied are documentation, book, report, letter, and note. If you want to edit these macros to change details of the styles, this should be much easier to do here than in LaTEX.

"MacroTEX can also be used to prepare a table of contents, lists of tables and figures, cross-references, indexes, glossaries, bibliographies, and end notes. Automatic equation numbering and cross-referencing is another feature. Variations of page parameters are easier to do than with Plain TEX.

"Amy Hendrickson has prepared an excellent manual with lots of illustrations. There is a great deal to be pleased with in this package."

A.G.W. Cameron, Professor of Astronomy, at the Harvard-Smithsonian Center for Astrophysics

Modular construction

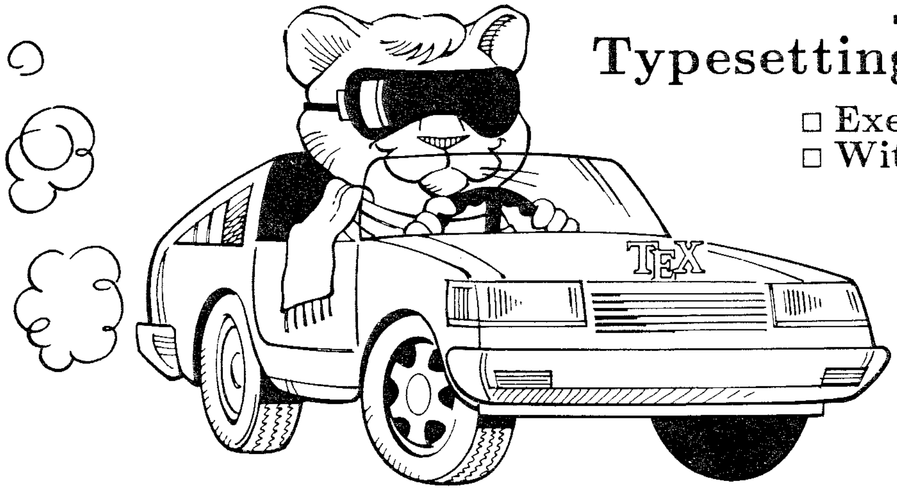
Flexibility

Source Code included

MacroTEX

Write or call for more information:

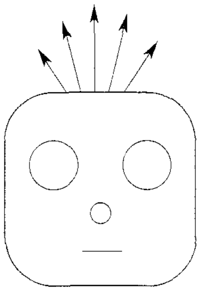
TEXnology Inc, 57 Longwood Avenue Brookline, MA 02146
(617) 738-8029


TEX

PiTeX

INTERACTIVE GRAPHICS

PICTURE YOURSELF
USING TeX



- *PiTeX is a graphics editor for use with LATeX*
- *PiTeX enables the novice to create and edit LATeX graphics in a WYSIWYG environment*
- *PiTeX offers the overworked TeX user full compatibility with LATeX*
- *PiTeX provides the connoisseur with speed, versatility and optimized performance*

PiTeX has two components:

PITEX - the interactive picture editor

PITRANS - the translator to LATeX picturemode commands

PiTeX runs on any IBM PC, PS/2 and compatibles with graphics

PiTeX is available now worldwide in several languages

US regular list price: $ 195 *Orders received before Sept 1: $ 135*

To order in the US: Send check or money order to our US branch. Add $ 5 for handling and shipping. MA residents add tax.

To order elsewhere: Ask your local PCTeX dealer.

INTERBASE

Europe's complete TeX source

INTERBASE USA INC.
ONE KENDALL SQ.
BUILDING 600, SUITE 138
CAMBRIDGE, MA 02139
(617) 643-4008

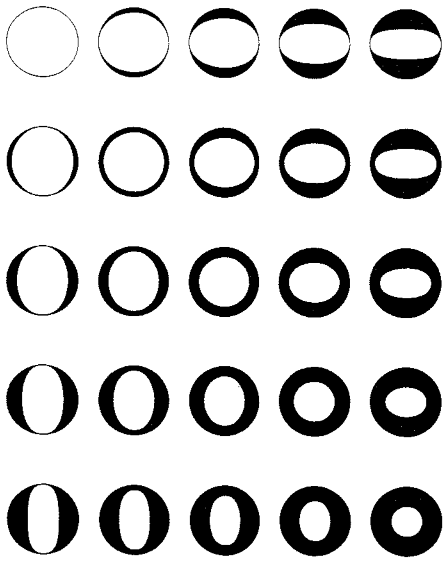
INTERBASE
DANTES PLADS 1
DK-1556 COPENHAGEN V
(01) 93 28 29
FAX: (01)93 36 93

TeX is a trademark of the AMS

IBM and PS/2 are trademarks of International Business Machines Corp.

PiTeX is a trademark of Interbase.

Index of Advertisers



Bugs in *Computers & Typesetting*

19 June 1988

This is a list of corrections made to *Computers & Typesetting*, Volumes A–E, since the last publication of the Errata and Changes list (15 February 88). Corrections to the softcover version of *The TEXbook* are the same as corrections to Volume A; corrections to the softcover version of *The METAFONTbook* are the same as corrections to Volume C.

Page A359, lines 35–38 (5/24/88)

```
\def\updownarrow{\delimiter"326C33F } \def\arrowvert{\delimiter"033C000 }
\def\Updownarrow{\delimiter"326D377 } \def\Arrowvert{\delimiter"033D000 }
\def\vert{\delimiter"026A30C }        \def\Vert{\delimiter"026B30D }
\def\backslash{\delimiter"026E30F }   \def\bracevert{\delimiter"033E000 }
```

Page A364, line 35 (6/19/88)

```
\def\fmtname{plain}\def\fmtversion{2.93} % identifies the current format
```

Page A463, left column (4/17/88)

*\day, 273, 349, *406*.

Page B2, line 32 (6/19/88)

```
define banner ≡ 'This␣is␣TeX,␣Version␣2.93'   { printed when TeX starts }
```

Page B54, line 35 (6/19/88)

```
else $t \leftarrow lo\_mem\_max+1+(hi\_mem\_min-lo\_mem\_max)$ div 2;   { $lo\_mem\_max+2 \le t < hi\_mem\_min$ }
```

Page B108, new line after line 8 (5/24/88)

```
d: integer;   { number of characters in incomplete current string }
```

Page B108, lines 31–33 (5/24/88)

```
str_room(l); $d \leftarrow cur\_length$;
while $pool\_ptr > str\_start[str\_ptr]$ do
  begin decr(pool_ptr); $str\_pool[pool\_ptr + l] \leftarrow str\_pool[pool\_ptr]$;
  end;   { move current string up to make room for another }
for $k \leftarrow j$ to $j + l - 1$ do append_char(buffer[k]);
$text(p) \leftarrow make\_string$; $pool\_ptr \leftarrow pool\_ptr + d$;
```

Page B115, line 12 (4/28/88)

```
group_code = 0 .. max_group_code;   { save_level for a level boundary }
```

Page B141, line 19 (4/28/88)

```
par_token: halfword;   { token representing '\par' }
```

Page B150, line 24 (4/28/88)

358. The present point in the program is reached only when the *expand* routine has inserted

Page B151, mini-index (4/28/88)

Delete the entry for '*no_expand*'; replace it by:
expand: **procedure**, §366.

Page B162, lines 12–14 (4/30/88)

```
repeat link(temp_head) ← null;
  if (info(r) > match_token + 127) ∨ (info(r) < match_token) then s ← null
  else begin match_chr ← info(r) − match_token; s ← link(r); r ← s; p ← temp_head; m ← 0;
```

Page B181, line 31 (4/28/88)

[Change 'x units per sp' to 'x sp per unit'! This change also should be made on line 1 of page B183 and line −8 of page B590.]

Page B188, line 8 (5/25/88)

```
function str_toks(b : pool_pointer): pointer; { changes the string str_pool[b .. pool_ptr] to a token list }
```

Page B188, line 13 (5/25/88)

```
begin str_room(1); p ← temp_head; link(p) ← null; k ← b;
```

Page B188, line 20 (5/25/88)

```
pool_ptr ← b; str_toks ← p;
```

Page B188, new line after line 28 (5/25/88)

```
b: pool_pointer; { base of temporary string }
```

Page B188, line 31 (5/25/88)

```
else begin old_setting ← selector; selector ← new_string; b ← pool_ptr;
```

Page B188, line 41 (5/25/88)

```
selector ← old_setting; the_toks ← str_toks(b);
```

Page B190, lines 16–18 (5/25/88)

```
b: pool_pointer; { base of temporary string }
begin c ← cur_chr; ⟨Scan the argument for command c 471⟩;
old_setting ← selector; selector ← new_string; b ← pool_ptr; ⟨Print the result of command c 472⟩;
selector ← old_setting; link(garbage) ← str_toks(b); ins_list(link(temp_head));
```

Page B210, line 36 (5/25/88)

```
begin if (pool_ptr + name_length > pool_size) ∨ (str_ptr = max_strings) ∨ (cur_length > 0) then
```

Page B285, line 21 (4/28/88)

is subsidiary to the *nucleus* field of some noad; the dot is replaced by '_' or '^' or '/' or '\' if p is

Page B387, line 2 (5/24/88)

is quite short. In the following code we set $hc[hn+2]$ to the impossible value 128, in order to

Page B387, line 8 (5/24/88)

```
$hc[0] \leftarrow 127$; $hc[hn+1] \leftarrow 127$; $hc[hn+2] \leftarrow 128$;   { insert delimiters }
```

Page B390, lines 17–18 (5/24/88)

⟨ Enter as many hyphenation exceptions as are listed, until coming to a right brace; then **return** 961 ⟩;

[The same change applies to lines 20–21, and to page 582.]

Page B396, new line after line 34 (5/24/88)

```
$trie\_link(trie\_size) \leftarrow 0$; $trie\_back(0) \leftarrow trie\_size$;   { wrap around }
```

Page B397, lines 15–17 (5/24/88)

```
begin $c \leftarrow trie\_c[p]$;
if $c < trie\_min$ then $trie\_min \leftarrow c$;
if $trie\_min = 0$ then $z \leftarrow trie\_link(trie\_size)$
else $z \leftarrow trie\_link(trie\_min - 1)$;   { get the first conceivably good hole }
```

Page B400, lines 3–4 (5/24/88)

⟨ Enter all of the patterns into a linked trie, until coming to a right brace 961 ⟩ ≡

[The same change applies to page B399, lines 29–30, and to page 582.]

Page B402, line 10 (5/24/88)

```
$r \leftarrow trie\_size$;   { finally, we will zero out the holes }
```

Page B552, left column (4/28/88)

[Insert '358' into *expand*.]

Page B562, left column (4/28/88)

[Remove '358' from *no_expand*.]

Page B568, left column (4/28/88)

[Move '269' from *save_index* to *save_level*.]

Page C164, line 10 (4/27/88)

```
$y_{\$c} = top\ y_{\$l}$;  $y_{\$d} = y_{\$r}$;  $x_{\$c} = x_{\$l} - left\_jut$;  $x_{\$d} = x_{\$r} + right\_jut$;
```

Page C241, line 11 (5/25/88)

```
numeric $ht^\#, dp^\#$;  $ht^\# = body\_height^\#$;  $.5[ht^\#, -dp^\#] = axis^\#$;
```

Page C337, line 11 (4/28/88)

An online "menu" of the available test routines will be typed at your terminal

Page D2, line 27 (6/19/88)

```
define banner ≡ 'This␣is␣METAFONT,␣Version␣1.5'   { printed when METAFONT starts }
```

Page D66, line 35 (6/19/88)

```
else $t \leftarrow lo\_mem\_max+1+(hi\_mem\_min-lo\_mem\_max)$ div 2;   { $lo\_mem\_max+2 \le t < hi\_mem\_min$ }
```

Page D420, bottom line (5/25/88)

```
    if txx mod unity = 0 then
```

Page D441, delete line 2 and change line 12 as follows (5/25/88)

```
done: if eq_type(x) ≠ tag_token then clear_symbol(x, false);
  if equiv(x) = null then new_root(x);
  scan_declared_variable ← h;
```

Changes to the Programs and Fonts

19 June 1988

TEX

Changes subsequent to errata publication, 15 February 88:

```
338. \outer\def\a0{}\a\a shows temp_head list garbage (Silvio Levy, 20Apr88)
@x module 391
repeat if (info(r)>match_token+127)or(info(r)<match_token) then s:=null
else  begin match_chr:=info(r)-match_token; s:=link(r); r:=s;
  p:=temp_head; link(p):=null; m:=0;
@y
repeat link(temp_head):=null;
if (info(r)>match_token+127)or(info(r)<match_token) then s:=null
else  begin match_chr:=info(r)-match_token; s:=link(r); r:=s;
  p:=temp_head; m:=0;
@z
```

```
339. \def\\#1{}\input a\\b failed (Robert Messer, 24Apr88)
@x module 259
var h:integer; {hash code}
@y
var h:integer; {hash code}
@!d:integer; {number of characters in incomplete current string}
@z
@x module 260
str_room(l);
for k:=j to j+l-1 do append_char(buffer[k]);
text(p):=make_string;
@y
str_room(l); d:=cur_length;
while pool_ptr>str_start[str_ptr] do
  begin decr(pool_ptr); str_pool[pool_ptr+l]:=str_pool[pool_ptr];
  end; {move current string up to make room for another}
for k:=j to j+l-1 do append_char(buffer[k]);
text(p):=make_string; pool_ptr:=pool_ptr+d;
@z
```

340. Make patterns work when trie_min=0 (Peter Breitenlohner, 10May88)

```
@x module 951
trie_max:=128; trie_min:=128; trie_link(0):=1; trie_taken[0]:=false;
@y
trie_max:=128; trie_min:=128; trie_link(0):=1; trie_taken[0]:=false;
trie_link(trie_size):=0; trie_back(0):=trie_size; {wrap around}
@z

@x module 953
begin c:=trie_c[p]; {we have |c>0|}
if c<trie_min then trie_min:=c;
z:=trie_link(trie_min-1); {get the first conceivably good hole}
@y
begin c:=trie_c[p];
if c<trie_min then trie_min:=c;
if trie_min=0 then z:=trie_link(trie_size)
else z:=trie_link(trie_min-1); {get the first conceivably good hole}
@z
@x module 966
r:=0; {finally, we will zero out the holes}
@y
r:=trie_size; {finally, we will zero out the holes}
@z
```

341. Avoid possible trie_pointer out of range (24May88)

```
@x module 923
hc[0]:=127; hc[hn+1]:=127; hc[hn+2]:=256; {insert delimiters}
@y
hc[0]:=127; hc[hn+1]:=127; hc[hn+2]:=128; {insert delimiters}
@z
```

342. \input a\romannumeral1 etc.; similar to bug 339. (25May88)

```
@x module 464
@p function str_toks:pointer; {changes the current string to a token list}
@y
@p function str_toks(@!b:pool_pointer):pointer;
  {changes the string |str_pool[b..pool_ptr]| to a token list}
@z
@x module 464, continued
p:=temp_head; link(p):=null; k:=str_start[str_ptr];
@y
p:=temp_head; link(p):=null; k:=b;
@z
@x module 464, concluded
pool_ptr:=str_start[str_ptr]; str_toks:=p;
@y
pool_ptr:=b; str_toks:=p;
@z
```

```
@x module 465
begin get_x_token; scan_something_internal(tok_val,false);
if cur_val_level>=ident_val then @<Copy the token list@>
else begin old_setting:=selector; selector:=new_string;
@y
@!b:pool_pointer; {base of temporary string}
begin get_x_token; scan_something_internal(tok_val,false);
if cur_val_level>=ident_val then @<Copy the token list@>
else begin old_setting:=selector; selector:=new_string; b:=pool_ptr;
@z

@x module 465, continued
  selector:=old_setting; the_toks:=str_toks;
@y
  selector:=old_setting; the_toks:=str_toks(b);
@z
@x module 470
begin c:=cur_chr; @<Scan the argument for command |c|@>;
old_setting:=selector; selector:=new_string;
@<Print the result of command |c|@>;
selector:=old_setting; link(garbage):=str_toks; ins_list(link(temp_head));
@y
@!b:pool_pointer; {base of temporary string}
begin c:=cur_chr; @<Scan the argument for command |c|@>;
old_setting:=selector; selector:=new_string; b:=pool_ptr;
@<Print the result of command |c|@>;
selector:=old_setting; link(garbage):=str_toks(b); ins_list(link(temp_head));
@z

343. **\input\romannumeral6 confusion bypassed (25May88)
@x module 525
begin if (pool_ptr+name_length>pool_size)or(str_ptr=max_strings) then
@y
begin if (pool_ptr+name_length>pool_size)or(str_ptr=max_strings)or
 (cur_length>0) then
@z

344. Avoid negative divisor rounding upward (Chris Thompson, fixed 19Jun88)
@x module 126
else t:=(lo_mem_max+hi_mem_min+2) div 2; {|lo_mem_max+2<=t<hi_mem_min|}
@y
else t:=lo_mem_max+1+(hi_mem_min-lo_mem_max)div 2; {|lo_mem_max+2<=t<hi_mem_min|}
@z

345. (I sincerely hope that there won't be any more)
```

METAFONT

Changes subsequent to publication of errata list, 15 February 88:

```
540. Typo suppresses an error detection (Chris Thompson, 2May88)
@x module 963
 if txy mod unity=0 then if tyy mod unity=0 then
@y
 if txx mod unity=0 then if tyy mod unity=0 then
@z

541. get_x_token can lose a scanned declared variable (Chris Thompson, 4May88)
@x module 1011
if equiv(x)=null then new_root(x);
@y
@z
@x module 1011
done:scan_declared_variable:=h;
@y
done: if eq_type(x)<>tag_token then clear_symbol(x,false);
if equiv(x)=null then new_root(x);
scan_declared_variable:=h;
@z

542. Avoid negative divisor rounding upward (Chris Thompson, fixed 19Jun88)
@x module 168
else t:=(lo_mem_max+hi_mem_min+2) div 2; {|lo_mem_max+2<=t<hi_mem_min|}
@y
else t:=lo_mem_max+1+(hi_mem_min-lo_mem_max)div 2; {|lo_mem_max+2<=t<hi_mem_min|}
@z

543. (I sincerely hope that there won't be any more)
```

Computer Modern fonts

Changes subsequent to publication of the errata list, 15 February 88:

No later changes.

TEX Users Group Membership List — Supplement

August 1988

This supplementary list, compiled on 16 June 1988, includes the names of all persons who have become members of TUG or whose addresses have changed since publication of the full membership list, compiled on 15 February 1988 and bound into TUGboat Vol. 9, No. 1. All institutional members are listed. Total membership: 149 institutional members and 2,886 individuals affiliated with more than 1,350 colleges and universities, commercial publishers, government agencies, and other organizations throughout the world having need for an advanced composition system.

The following information is included for each listing of an individual member, where it has been provided:

Name and mailing address

Telephone number

Network address

Title and organizational affiliation, when that is not obvious from the mailing address

Computer and typesetting equipment available to the member, or type of equipment on which his organization wishes to (or has) installed TEX

Uses to which TEX may be put, or a general indication of why the member is interested in TEX

CONTENTS

Recipients of this list are encouraged to use it to identify others with similar interests, and, as TUG members, to keep their own listings up-to-date in order for the list to remain as useful as possible. New or changed information may be submitted on the membership renewal form bound into the back of a recent issue of TUGboat. Comments on ways in which the content and presentation of the membership list can be improved are welcome.

This list is intended for the private use of TUG members; it is not to be used as a source of names to be included in mailing lists or for other purposes not approved by TUG. Additional copies are available from TUG. Mailing lists of current TUG membership are available for purchase. For more information, contact Ray Goucher, TUG Executive Director.

Distributed with TUGboat Volume 9 (1988), No. 2. Published by

TEX Users Group
P. O. Box 9506
Providence, R.I. 02940-9506, U.S.A.

Talaris Systems, Inc.
San Diego, California

Texas A & M University
Computing Services Center
College Station, Texas

Texas A & M University
Department of Computer Science
College Station, Texas

Tribune TV Log
Glens Falls, New York

TRW, Inc.
Redondo Beach, California

Tufts University
Medford, Massachusetts

TV Guide
Radnor, Pennsylvania

TYX Corporation
Reston, Virginia

UNI.C
Danmarks EDB-Center
Aarhus, Denmark

University College
Cork, Ireland

University of Alabama
Tuscaloosa, Alabama

University of British Columbia
Computing Centre
Vancouver, British Columbia, Canada

University of British Columbia
Mathematics Department
Vancouver, British Columbia, Canada

University of Calgary
Calgary, Alberta, Canada

University of California, Berkeley
Academic Computing Services
Berkeley, California

University of California, Berkeley
Computer Science Division
Berkeley, California

University of California, Irvine
Department of Mathematics
Irvine, California

University of California, Irvine
Information & Computer Science
Irvine, California

University of California, San Diego
La Jolla, California

University of California, San Francisco
San Francisco, California

University of Chicago
Computation Center
Chicago, Illinois

University of Chicago
Computer Science Department
Chicago, Illinois

University of Chicago
Graduate School of Business
Chicago, Illinois

University of Crete
Institute of Computer Science
Research Center
Heraklio, Crete, Greece

University of Delaware
Newark, Delaware

University of Exeter
Computer Unit
Exeter, Devon, England

University of Glasgow
Glasgow, Scotland

University of Groningen
Groningen, The Netherlands

University of Illinois at Chicago
Computer Center
Chicago, Illinois

University of Illinois at Urbana-Champaign
Computer Science Department
Urbana, Illinois

University of Kansas
Academic Computing Services
Lawrence, Kansas

University of Maryland
College Park, Maryland

University of Massachusetts
Amherst, Massachusetts

University of North Carolina
School of Public Health
Chapel Hill, North Carolina

University of Oslo
Institute of Informatics
Blindern, Oslo, Norway

University of Ottawa
Ottawa, Ontario, Canada

University of Southern California
Information Sciences Institute
Marina del Rey, California

University of Stockholm
Department of Mathematics
Stockholm, Sweden

University of Tennessee at Knoxville
Department of Electrical Engineering
Knoxville, Tennessee

University of Texas at Austin
Physics Department
Austin, Texas

University of Texas at Dallas
Center for Space Science
Dallas, Texas

University of Vermont
Burlington, Vermont

University of Washington
Department of Computer Science
Seattle, Washington

University of Western Australia
Regional Computing Centre
Nedlands, Australia

University of Wisconsin
Academic Computing Center
Madison, Wisconsin

Uppsala University
Uppsala, Sweden

Vanderbilt University
Nashville, Tennessee

Vereinigte Aluminium-Werke AG
Bonn, Federal Republic of Germany

Villanova University
Villanova, Pennsylvania

Vrije Universiteit
Amsterdam, The Netherlands

Washington State University
Pullman, Washington

Widener University
Computing Services
Chester, Pennsylvania

John Wiley & Sons, Incorporated
New York, New York

Worcester Polytechnic Institute
Worcester, Massachusetts

Yale University
Computer Center
New Haven, Connecticut

Yale University
Department of Computer Science
New Haven, Connecticut

(149 institutional members
as of 1 Jul 1988)

TeX Consulting and Production Services

North America

ALDINE PRESS
12625 La Cresta Drive, Los Altos Hills, CA 94022;
(415) 948-2149
Composition services for mathematical and technical books.

AMERICAN MATHEMATICAL SOCIETY
P. O. Box 6248, Providence, RI 02940;
(401) 272-9500, ext. 224
Typesetting from DVI files on an Autologic APS Micro-5. Times Roman and Computer Modern fonts. Composition services for mathematical and technical books and journal production.

ARBORTEXT, Inc.
535 W. William, Suite 300, Ann Arbor, MI 48103;
(313) 996-3566
Typesetting from DVI files on an Autologic APS-5. Computer Modern and standard Autologic fonts. TeX installation and applications support. TeX-related software products.

ARCHETYPE PUBLISHING, Inc.,
Lori McWilliam Pickert
P. O. Box 6567, Champaign, IL 61821; (217) 359-8178
Experienced in producing and editing technical journals with TeX; complete book production from manuscript to camera-ready copy; TeX macro writing including complete macro packages; consulting.

HOENIG, Alan
17 Bay Avenue, Huntington, NY 11743; (516) 385-0736
TeX typesetting services including complete book production; macro writing; individual and group TeX instruction.

KUMAR, Romesh
1028 Emerald, Naperville, IL 60540; (312) 972-4342
Beginners and intermediate group/individual instruction in TeX. Development of TeX macros for specific purposes. Using TeX with FORTRAN for custom-tailored software. Flexible hours, including evenings and weekends.

OGAWA, Arthur
920 Addison, Palo Alto, CA 94301; (415) 323-9624
Experienced in book production, macro packages, programming, and consultation. Complete book production from computer-readable copy to camera-ready copy.

RICHERT, Norman
1614 Loch Lake Drive, El Lago, TX 77586;
(713) 326-2583
TeX macro consulting.

TeXNOLOGY, Inc., Amy Hendrickson
57 Longwood Ave., Brookline, MA 02146;
(617) 738-8029.
TeX macro writing (author of MacroTeX); custom macros written to meet publisher's or designer's specifications; instruction.

Outside North America

BAZARGAN, Kaveh
Optics Section, Blackett Laboratory, Imperial College
of Science and Technology, London SW7 2BZ, U.K.;
(01) 589 5111 ext. 6841
Instruction in TeX, for beginner and intermediate levels. Custom macros, including complex tables.

TeXPERT SYSTEMS Ltd.,
(Malcolm Clark and Cathy Booth)
5, Northernhay Square, Exeter, EX4 3ES, Devon, U.K.;
(0392) 76091
The Complete TeX Service: consultation, all levels of document production, printing, customised style sheets (macros) to simplify text input, comprehensive training courses (will travel).

TeXWORKS Pty. Ltd.
157 Danks Street, Albert Park, Melbourne, Victoria 3206,
Australia; 61 3 699 4083
Commercial versions of TeX and screen previewers for computers including PC, Macintosh, Amiga, Unix, VAX, and TeX-related products; print drivers for dot matrix printers, 300 dpi lasers, and phototypesetters (Autologic, Compugraphic and Linotronic); assistance with information on public domain software; consulting for typesetters and other TeX users.

VARDAS
88, North Rd., St. Andrews, Bristol, BS6 5AJ, U.K.;
(0272) 428709
TeX consultation, instruction, macro writing, book production, consulting.

Information about this service can be obtained from Ray Goucher, TeX Users Group, P. O. Box 9506, Providence, RI 02940, (401) 751-7760.