a ship of thought, deep freighted with truth, with Beauty, too.
Typologia, Frederic W. Goudy

TUGBOAT

Communications of the TEX Users Group

Editor Barbara Beeton
Guest Editors David Kellerman and Barry Smith

Volume 7, Number 1 March 1986
Providence, Rhode Island U.S.A.

TUGBOAT

The communications of the TEX Users Group are published irregularly at Providence, Rhode Island, and are distributed as a benefit of membership both to 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. 7, No. 2, is April 28, 1986; the issue will be mailed in late June.

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 tape or as camera-ready copy should be addressed to the Editor, Barbara Beeton.

Contributions in camera copy form are encouraged, as is electronic submission of items on magnetic tape, via electronic mail, or transferred directly to the AMS computer; for instructions, write or call Barbara Beeton.

TUGBOAT Advertising and Mailing Lists

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

TUGBOAT

Volume 7, Number 1
March 1986

ADDRESSES

Wolfgang Appelt
Gesellschaft für Mathematik und Datenverarbeitung
1000 Schloss Birlinghoven - PF 1240
D-5202 St Augustin 1
Federal Republic of Germany

Alan Auerbach
Wilfrid Laurier University
Waterloo N2L 3C5 Ontario
Canada

Bruce W. Baker
Textset
P.O. Box 7993
Ann Arbor, MI 48107
313-996-3566

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
Arpanet: `bb@SU-AI`

Neil Block
P.O. Box 8829
Fountain Valley, CA 92728-8829
213-513-4545

Lance Carnes
163 Linden Lane
Mill Valley, CA 94941
415-388-8853

S. Bart Childs
Department of Computer Science
Texas A & M University
College Station, TX 77843-3112
409-845-5470

Maria Code
Data Processing Services
1371 Sydney Dr
Sunnyvale, CA 94087

John M. Crawford
Computing Services Center
College of Administrative Science
Ohio State University
Columbus, OH 43210
614-422-1741
CSNet: `Crawford-J@Ohio-State`
BITNet: `TS0135@OHSTVMA`

R. M. Damerell
Mathematics Department
Royal Holloway-Bedford College
Egham
Surrey, England
07843-4455
Arpanet: `Damerell@UCL-CS`

Jacques Désarménien
Laboratoire de typographie informatique
Université Louis-Pasteur
7, rue René-Descartes
67084 Strasbourg Cedex, France

Chuck Dupree
Omnicad Corp
1000 Pittsford-Victor Road
Pittsford, NY 14534
716-385-8500

Maureen Eppstein
Administrative Publication Manager
Stanford University
Encina Hall, Room 200
Stanford, CA 94305
415-497-9254
Arpanet: `MVEppstein@SU-Score`

Michael J. Ferguson
INRS - Télécommunications
Université du Québec
3 Place du Commerce
Verdun H3E 1H6 Québec
Canada
514-765-7834

Patricia M. Fina
333 Columbus Avenue #31
Boston, MA 02116
617-253-8449

Jim Fox
Academic Computing Center HG-45
University of Washington
3737 Brooklyn Avenue NE
Seattle, WA 98105
206-543-4320
Bitnet: `fox7632@uwacdc`

David Fuchs
Department of Computer Science
Stanford University
Stanford, CA 94305
415-497-1646
Arpanet: `DRF@SU-Score`

Richard Furuta
Department of Computer Science
University of Maryland
College Park, MD 20742
301-454-1461
Arpanet: `Furuta@Maryland`

Raymond E. Goucher
TEX Users Group
P.O. Box 9506
Providence, RI 02940
401-272-9500 x232

William Gropp
Department of Computer Science
Yale University
Box 2158 Yale Station
New Haven, CT 06520
203-436-3761
Arpanet: `Gropp@Yale`

Alan Hoenig
574 Argyle Road
Brooklyn, NY 11230
718-856-3696

Karin Horn
Gesellschaft für Mathematik und Datenverarbeitung
Schloss Birlinghoven - PF 1240
D-5202 St Augustin 1
Federal Republic of Germany

Gordon K. Howell
Hughes Aircraft Company
P.O. Box 92919
Bldg 550/X303
Los Angeles, CA 90009
213-648-8994
`engvax!gordon@CIT-VAX.ARPA`

Patrick D. Ion
Mathematical Reviews
416 Fourth Street
P.O. Box 8604
Ann Arbor, MI 48107
313-763-6829

Helmut Jürgensen
Department of Computer Science
University of Western Ontario
London N6A 5B7 Ontario
Canada
519-679-3039
Bitnet: `A505@UWOCC1`
UUCP: `helmut@deepthot`

Arthur Keller
University of Texas at Austin
Department of Computer Science
Austin, TX 78712-1188
512-471-7316
Arpanet: `ARK@SALLY.UTexas.EDU`

David Kellerman
Kellerman & Smith
534 SW Third Avenue
Portland, OR 97204
503-222-4234

Bill Kelly
Academic Computing Center
University of Wisconsin, Madison
1210 W. Dayton Street
Madison, WI 53706
608-262-9501
Arpanet:
`uwmacc!bllklly@rsch.wisc.edu`

Donald E. Knuth
Department of Computer Science
Stanford University
Stanford, CA 94305
Arpanet: `DEK@SU-AI`

Leslie Lamport
Systems Research Center
Digital Equipment Corp
130 Lytton Avenue
Palo Alto, CA 94301
415-853-2170
Arpanet: `lamport@decwrl`

Pierre A. MacKay
University of Washington
Department of Computer Science
FR-35
Seattle, WA 98195
206-543-2386
Arpanet: `MacKay@Washington`

Rick Mallett
Computing Services
Room 1208 Arts Tower
Carleton University
Ottawa K1S 5B6 Ontario
Canada
613-231-7145

Doug Maus
Textset
P. O. Box 7993
Ann Arbor, MI 48107
313-996-3566

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

Mitch Pfeffer
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
Department of Math & Astronomy
Machray Hall
University of Manitoba
Winnipeg R3T 2N2 Manitoba
Canada
204-474-9832
`platt%uofm-uts.cdn@ubc.csnet`

Mark Senn
107 Digby Road
Lafayette, IN 47905-1163
317-743-0560

Barry Smith
Kellerman & Smith
534 SW Third Avenue
Portland, OR 97204
503-222-4234
`...!tektronix!reed!barry`

Alan Spragens
SLAC Computing Services
Stanford Linear Accelerator Center
Bin 97, P. O. Box 4349
Stanford, CA 94305
415-854-3300 x2849
Bitnet: `Spragens@SLACVM`

Jim Sterken
Textset
P. O. Box 7993
Ann Arbor, MI 48107
313-971-3628

Ralph Stromquist
MACC
University of Wisconsin
1210 W. Dayton Street
Madison, WI 53706
608-262-8821

Rilla Thedford
Intergraph Corporation
One Madison Industrial Park
Huntsville, AL 35807
205-772-6494

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

Andrew Trevorrow
c/o Computing Centre
University of Adelaide
Box 498, G P O, Adelaide
South Australia 5001
Australia
(08) 228 5984

Joey K. Tuttle
I P Sharp Associates
220 California Avenue
Suite 201
Palo Alto, CA 94306
415-327-1700

Hal R. Varian
Department of Economics
University of Michigan
Ann Arbor, MI 48109
313-764-2364

Samuel B. Whidden
American Mathematical Society
P. O. Box 6248
Providence, RI 02940
401-272-9500

Hermann Zapf
Seitersweg 35
D-6100 Darmstadt
Federal Republic of Germany

Calendar

April 14–16, 1986	International Conference on Text Processing and Document Manipulation	University of Nottingham, England
April 28, 1986	*TUGBOAT* 7, no. 2, deadline for submission of manuscripts	Barbara Beeton, TEX Users Group
June 19–21, 1986	TEX for Scientific Documentation announcement on page 8	Jacques Désarménien, Strasbourg, France
July 14–15, 1986	Intermediate TEX Course	Tufts University, Medford, Massachusetts
July 16–18, 1986	Advanced TEX Course	Tufts University, Medford, Massachusetts
July 21–23, 1986	TEX Users Group Annual Meeting	Tufts University, Medford, Massachusetts
July 24–25, 1986	TEX Short Course on Output Routines	Tufts University, Medford, Massachusetts
August 18–22, 1986	ACM SIGGRAPH 86	Dallas, Texas
August 25, 1986	*TUGBOAT* 7, no. 3, deadline for submission of manuscripts	Barbara Beeton, TEX Users Group
October 20–24, 1986	PROTEXT III	Dublin, Ireland

TUG News

From the President

I carefully read the November issue of *TUGBOAT* and was shocked to see that I did not acknowledge the contributions of Pierre MacKay, Michael Spivak, and the founders of our organization. I correct this by thanking them and the AMS for creation of a good productive organization. Organizations like TUG always depend upon some good people contributing of their time. These people have really set a solid foundation for us to build upon.

Michael Spivak has resigned from the Steering Committee since he is now strictly in the TeX community for financial gain. I am sure that he will continue to be available as a sounding board.

Nelson Beebe, John Gourlay, and Richard Palais have agreed to be the nominating committee for selecting a candidate (or slate) for vice president and for secretary at the Tufts meeting.

I have given TeX talks to sister institutions, a seminar at the Joint Math meeting in New Orleans, a shortcourse (one day) after the meeting, and have more presentations of these types scheduled for the future.

Jim Fox at the University of Washington's Academic Computing Center has become the CDC Cyber site coordinator.

I continue to be pleased with the staff of TUG. Ray, Barbara, and Sam continue to have my complete support. We are fortunate.

Will each site coordinator please send me an indication of: (1) The contents (and levels) of their TeX distribution; (2) The source (or listing) of their *Local Guide* as Leslie Lamport calls it in the LaTeX manual.

One of my goals is to have a generic form and minimum levels for all. Happy TeXing.

Bart Childs
Texas A & M University

Acknowledgment of Contributions

The Officers and Steering Committee gratefully acknowledge receipt of royalties and other contributions to TUG from several sources during 1985:

Don Knuth From the sale of Don Knuth's *The TeXbook*, royalties of $3,787.

Kellerman & Smith From David Kellerman and Barry Smith, of Kellerman & Smith, royalties of $1,325, from fees paid to them for distribution of the VAX/VMS version of TeX.

Arthur Samuel More than 800 copies of Arthur Samuel's *First Grade TeX* sold by TUG in 1985 resulted in the addition of $5,900 to TUG's treasury.

Textset Contributions of $1,500 received from David Rodgers on behalf of Textset, Inc.

TUG sincerely appreciates these very generous contributions.

Samuel B. Whidden
Treasurer, TeX Users Group

Donald E. Knuth Scholarship

TUG is pleased to announce the first annual "Donald E. Knuth Scholarship." The award consists of an all-expense-paid trip to TUG's 1986 Annual Meeting and the two-day course offered immediately following the meeting. Competition for the scholarship is open to all 1986 TUG members holding secretarial or clerical support staff positions.

To enter the competition, applicants should submit to the Scholarship Committee by May 15, 1986, the input file and final TeX output of a project that displays originality, knowledge of TeX, and good TeXnique. Along with the TeX files, each applicant should submit a letter stating that he/she will be able to attend the Annual Meeting and Short Course at Tufts University, Medford, Massachusetts, July 21–25, 1986. This year's course will be on TeX Output Routines.

Selection of the scholarship recipient will be based on the TeX sample. Judging will take place May 16–June 15, and the winner will be notified by mail after June 16.

The Scholarship Committee consists of Maria Perkins, Vanderbilt University, chairperson; Carl Smith, University of Maryland; and Linda Woessner, Computer Science Corp. Funding for the first year of this scholarship will be provided by Kellerman & Smith of Portland, Oregon.

All applications should be submitted to the Committee at the following address:

Maria Perkins
Computer Center
Vanderbilt University
Box 1577 Station B
Nashville, TN 37235

Special Credits for This Issue

This issue of *TUGBOAT* is, more than usual, an experiment. First of all, it has been guest edited, by David Kellerman and Barry Smith.

David and Barry originally volunteered for this task at the 1984 TUG meeting. Their intention was to use an issue of *TUGBOAT* to show how TEX works in the hands of a professional graphics designer. This vision has now become real.

The layout for this issue was designed by Martha Gannett, a designer based in Portland, Oregon. You may have seen her work before — her recent projects include the 1985 Floating Point Systems annual report, and the 1984, 1985, and 1986 Mentor Graphics annual reports. Her special fondness for book design, though, has resulted in some of her most memorable projects, including *Qiñiqtuagaksrat Utuqqanaat Iñuuniaġninisiqun*, a bilingual text of interviews with Innuipiat Indians. I am sure this project will also be among the memorable.

The macros have been realized by David, based on the existing *TUGBOAT* macros, at least to the extent that the input files of individual articles look virtually the same as they would have, had this issue been prepared in the usual way.

Pierre MacKay worked long and hard with David to install the new Computer Modern fonts on the Alphatype CRS at the University of Washington. Considering that only the old METAFONT contains the instructions to generate Alphatype fonts, this was no mean task: It involved generation of .GF files at 5333 dots per inch with METAFONT84, translating these images to a binary format, and recycling them through the old SAIL METAFONT. (I haven't counted the number of fonts that have been given this treatment, but I know it took at least a day's worth of computer time.)

Finally, camera copy was prepared on the Washington Alphatype and shipped to Providence for printing.

After they have had a suitable rest, I shall expect David and Barry to write up their experience, along with the new macros, for publication in a future issue. In the meantime, thanks and congratulations to all.

Barbara Beeton
American Mathematical Society

Conference: TEX for Scientific Documentation

The TEX for Scientific Documentation conference will take place 19–21 June 1986 at:

Centre culturel Saint-Thomas
2, rue de la Carpe-Haute
67000 Strasbourg, France

The aim of this conference is to provide a state-of-the-art survey of current work in this area, and to encourage technology transfer and information exchange on the latest applications of documentation systems based on TEX.

These institutions are sponsors of the conference: CNRS (Centre national de la Recherche scientifique), SMF (Societé mathématique de France), University Louis-Pasteur de Strasbourg.

Applications for registration must reach the Conference Chairman before 1 April 1986. Registration forms may be obtained from him or from the TUG office; instructions to authors will be included.

The Proceedings will be published in the Springer-Verlag Lecture Notes in Computer Science series. Contributions must be received by the Conference Chairman no later than 1 April 1986; the deadline for notification of acceptance has been postponed to 1 May 1986. Priority of publication will be given to contributions submitted by participants.

Jacques Désarménien
Conference Chairman
Laboratoire de typographie informatique
Université Louis-Pasteur
7, rue René-Descartes
67084 Strasbourg Cedex
France

TEX Courses

The TEX Users Group is offering the following schedule of courses in all skill levels of TEX programming. Contact Ray Goucher, TEX Users Group, for registration information.

All 3-day courses start on Mondays and 2-day courses on Thursdays, except as follows: Beginning and Intermediate Courses at Stanford, Tuesday–Thursday and Friday–Saturday, respectively; Intermediate and Advanced at Tufts, Monday–Tuesday and Wednesday–Friday, respectively.

A special series of courses [*150, 250, 306, 405*] for technical users with programming or scientific backgrounds has been scheduled to coincide with the completion on Monday, August

Special Credits for This Issue

This issue of *TUGBOAT* is, more than usual, an experiment. First of all, it has been guest edited, by David Kellerman and Barry Smith.

David and Barry originally volunteered for this task at the 1984 TUG meeting. Their intention was to use an issue of *TUGBOAT* to show how TEX works in the hands of a professional graphics designer. This vision has now become real.

The layout for this issue was designed by Martha Gannett, a designer based in Portland, Oregon. You may have seen her work before — her recent projects include the 1985 Floating Point Systems annual report, and the 1984, 1985, and 1986 Mentor Graphics annual reports. Her special fondness for book design, though, has resulted in some of her most memorable projects, including *Qiñiqtuagaksrat Utuqqanaat Iñuuniaġninisiqun*, a bilingual text of interviews with Innuipiat Indians. I am sure this project will also be among the memorable.

The macros have been realized by David, based on the existing *TUGBOAT* macros, at least to the extent that the input files of individual articles look virtually the same as they would have, had this issue been prepared in the usual way.

Pierre MacKay worked long and hard with David to install the new Computer Modern fonts on the Alphatype CRS at the University of Washington. Considering that only the old METAFONT contains the instructions to generate Alphatype fonts, this was no mean task: It involved generation of `.GF` files at 5333 dots per inch with METAFONT84, translating these images to a binary format, and recycling them through the old SAIL METAFONT. (I haven't counted the number of fonts that have been given this treatment, but I know it took at least a day's worth of computer time.)

Finally, camera copy was prepared on the Washington Alphatype and shipped to Providence for printing.

After they have had a suitable rest, I shall expect David and Barry to write up their experience, along with the new macros, for publication in a future issue. In the meantime, thanks and congratulations to all.

Barbara Beeton
American Mathematical Society

Conference: TEX for Scientific Documentation

The TEX for Scientific Documentation conference will take place 19–21 June 1986 at:

Centre culturel Saint-Thomas
2, rue de la Carpe-Haute
67000 Strasbourg, France

The aim of this conference is to provide a state-of-the-art survey of current work in this area, and to encourage technology transfer and information exchange on the latest applications of documentation systems based on TEX.

These institutions are sponsors of the conference: CNRS (Centre national de la Recherche scientifique), SMF (Societé mathématique de France), University Louis-Pasteur de Strasbourg.

Applications for registration must reach the Conference Chairman before 1 April 1986. Registration forms may be obtained from him or from the TUG office; instructions to authors will be included.

The Proceedings will be published in the Springer-Verlag Lecture Notes in Computer Science series. Contributions must be received by the Conference Chairman no later than 1 April 1986; the deadline for notification of acceptance has been postponed to 1 May 1986. Priority of publication will be given to contributions submitted by participants.

Jacques Désarménien
Conference Chairman
Laboratoire de typographie informatique
Université Louis-Pasteur
7, rue René-Descartes
67084 Strasbourg Cedex
France

TEX Courses

The TEX Users Group is offering the following schedule of courses in all skill levels of TEX programming. Contact Ray Goucher, TEX Users Group, for registration information.

All 3-day courses start on Mondays and 2-day courses on Thursdays, except as follows: Beginning and Intermediate Courses at Stanford, Tuesday–Thursday and Friday–Saturday, respectively; Intermediate and Advanced at Tufts, Monday–Tuesday and Wednesday–Friday, respectively.

A special series of courses [*150, 250, 306, 405*] for technical users with programming or scientific backgrounds has been scheduled to coincide with the completion on Monday, August

11, 1986, of the International Congress of Mathematicians, Berkeley, California.

Beginning TEX A 3-day course for non-technical users: *[101]* March 3–5, Vanderbilt University, Nashville, TN; *[102]* March 24–26, Illinois Institute of Technology, Chicago, IL; *[103]* March 24–26, Harvard University, Cambridge, MA; *[104]* June 2–4, Vanderbilt University, Nashville, TN; *[105]* July 28-30, Illinois Institute of Technology, Chicago, IL.

Beginning TEX A 3-day course for technical users: *[150]* August 12–14, Stanford University, Stanford, CA.

Intermediate TEX A 2-day course for non-technical users: *[201]* March 6–7, Vanderbilt University, Nashville, TN; *[202]* March 27–28, Illinois Institute of Technology, Chicago, IL; *[203]* March 27–28, Harvard University, Cambridge, MA; *[204]* June 5–6, Vanderbilt University, Nashville, TN; *[205]* July 14–15, Tufts University, Medford, MA; *[206]* July 31–August 1, Illinois Institute of Technology, Chicago, IL.

Intermediate TEX A 3-day course for technical users: *[250]* August 15–16, Stanford University, Stanford, CA.

Advanced TEX A 3-day course: *[301]* April 7–9, Vanderbilt University, Nashville, TN; *[302]* April 28–30, Illinois Institute of Technology, Chicago, IL; *[303]* June 2–4, Illinois Institute of Technology, Chicago, IL; *[304]* July 16–18, Tufts University, Medford, MA; *[305]* August 4–6, Vanderbilt University, Nashville, TN; *[306]* August 18–20, Stanford University, Stanford, CA.

Macro Writing A 2-day course: *[401]* April 10-11, Vanderbilt University, Nashville, TN; *[402]* May 1–2, Illinois Institute of Technology, Chicago, IL; *[403]* June 5–6, Illinois Institute of Technology, Chicago, IL; *[404]* August 7–8, Vanderbilt University, Nashville, TN; *[405]* August 21–22, Stanford University, Stanford, CA.

Output Routines A 2-day course: *[425]* July 24–25, Tufts University, Medford, MA.

Notes There are many TEX users who are not TUG members, but have expressed an interest in these courses. Please help them get the word by posting this notice.

TUG members and individuals on TUG's mailing list will be sent registration information and additional course/meeting information routinely. Anyone else wishing to receive future mailings should contact the TUG office.

Organizations desiring in-house courses, tailored to their specific needs, or interested in hosting Regional Courses should contact the TUG office.

Ray Goucher
TEX Users Group

Software Column: Call for Papers

This time we have four papers on TEX and `WEB` related software problems!

W. Appelt and K. Horn show how to `KNIT` and `TWIST` your program rather than `TANGLE` and `WEAVE` it. Their `PATCHWORK` system — as they call it — allows one to have more than one change file for a program; and this is really quite useful when a program has to be adapted to severa environments at the same time. I know of and have been using another similar system (called `TIE`) which I got from Darmstadt University a few months ago. I wonder whether further systems addressing the same problem exist. A comparison and evaluation of the solution strategies might be interesting.

The problem of error diagnostics of `TANGLE` is studied in R. M. Damerell's paper. He proposes and has implemented a slight change to `TANGLE` which improves `TANGLE`'s error detection capability. This is quite an important point; I'd appreciate receiving information about other related work for publication in *TUGBOAT*.

The third paper — by A. Trevorrow — describes a very powerful preview system for TEX. It supports quite a variety of visual displays. The benchmarks are really impressive.

Finally, M. Senn gives a survey of some preview packages. In addition to this paper I've received a few letters informing me about the existence of several other preview packages. We will prepare an update about such systems for publication in the next issue. If you have a TEX previewer, please send me information about it.

During this academic year I had a blind student in my classes. As he had a Braille terminal much of our written communication was by computer mail. In particular, I would send him the TEX input files of assignments, and he would get them printed in Braille (the input files!). This was fine with plain text and simple formulae. But just imagine reading a complex `\halign` in Braille! I am considering to have him work on a

DVI-to-Braille driver as a project. I wonder, however, whether such a driver already exists somewhere. (A letter asking a related question was read at the TUG '84 meeting, and published in *TUGBOAT* 5, no. 2: 146.) I'd appreciate getting comments and hints concerning this problem.

I close with my usual appeal: If you have software problems, ideas, suggestions, implementations, or papers related to typesetting, TEX, and WEB for *TUGBOAT*, please do send them to me.

Helmut Jürgensen
University of Western Ontario

Addenda: The BCS Word Processor Review

The Boston Computer Society's IBM PC & Compatibles Technical Word Processor Report reviewed in *TUGBOAT* 6, no. 3: 146–149, has now been published, in the January 1986 issue of the *Notices of the American Mathematical Society*, pages 8–37. The camera copy was prepared using TEX and the Society's Alphatype CRS typesetter, except for one illustration of Troff input and output.

Copies of the complete report may be obtained by sending a check for $8 (Massachusetts residents should add 5% sales tax) to:

Carl A. Hein
Dunster House, Apartment 7
Swanson Road
Boxborough, MA 01719

The two individuals cited as authors of the report, Avram Tetewsky and Jack Pearson, wish it known that credit is also due to the other members of the review committee who tested the software and wrote the individual reviews. Herewith are the names of all the authors: Damon Bostick, MIT; A. G. W. Cameron, Harvard University; Jim Cronin, Yankee Atomic; Mike Fellows, MIT; Peter Ford, MIT; Richard Goldstein, Qualitas; William Gilbert, MIT; Chase Green, Raytheon; Robert Indik, Brandeis University; Jim Loomis, Yankee Atomic; Pete Matthews, Jr., C. S. Draper Laboratory; Jack Pearson, Avco Systems; Avram Tetewsky, C. S. Draper Laboratory.

The benchmark examples published in both *TUGBOAT* and the AMS *Notices* were not labeled on each page as originating in the BCS review. It is requested, if these pages are reproduced for any reason, that each copy be clearly marked to credit the PC Technical Group of the IBM PC Users Group of the Boston Computer Society.

Barbara Beeton
American Mathematical Society

LETTERS

LaTeX input to ACM journals

To the Editor:

I am in contact with ACM about the possibility of authors providing camera-ready copy or LaTeX input files to ACM journals, bypassing the typesetting/proofreading hassles involved in publication. I'm interested in finding a few authors who have articles, produced in LaTeX, that have been submitted to ACM journals and would like to be guinea pigs for a new procedure.

Authors interested in this experiment should communicate via the Arpanet.

Leslie Lamport
`Lamport@decwrl.DEC.COM`

Translators to Generate TeX files

To the Editor:

Does anybody have a program to translate RUNOFF or BOLIO source documents to TeX (or LaTeX)? These could be written in almost any language (including my favorites: TECO, SNOBOL, Lisp, and Scan).

I will settle for 95% solutions...

Gordon Howell
Hughes Aircraft Company
P. O. Box 92919
Bldg. 550/X303
Los Angeles, CA 90009
`engvax!gordon@CIT-VAX.ARPA`

Short Reports

News from the TeX Project

The news from Stanford is that METAFONT 1.0 is now officially released, along with the entire Computer Modern family. TeX 1.5, when used with the new CM fonts, is officially called TeX 2.0. Distribution tapes with all of this stuff should be done by the time you read this. Font tapes now come in GF format, and include more sizes and magnifications (as well as more different faces) than with the old AM series.

The five-volume series *Computers and Typesetting* is almost complete, and 90% of it has already gone to the printer as of this writing. Volume A is *The TeXbook*, B is the TeX program, C is *The METAFONTbook*, D is the METAFONT program, and E is about Computer Modern. *The TeXbook* and *The METAFONTbook* will both be available in softcover (like *The TeXbook* has been).

David Fuchs
Stanford University

CDC Cyber Site Report

This article introduces a new CDC TeX implementation. It has been running under NOS 2 on our Cyber 180-855 at the Academic Computing Center since the spring of 1985. Our first version was 1.1 but we upgraded to version 1.5 last fall. Printing is done by a pair of QMS Lasergrafix 1200 laser printers.

TeX is a big program and I made no attempt to overlay or split it in any way. We therefore usually run it as a batch job. This is not a problem since TeX formatting is essentially a batch process anyway. We also provide a small, online version that makes a useful learning tool, but is not capable of much real formatting. The batch TeX runs in about 370K (octally speaking); the online version takes about 230K.

Our printer driver (`PTeX`) is derived from `dvitype`. (My guess is that all printer drivers are derived from `dvitype`.) Ours has been extended to provide a very flexible and easy-to-use graphics insertion capability. Essentially, `PTeX` will scale, center, and draw a 'real' plot (e.g., a Calcomp plot), without the need for any custom fitting by the user. A `\special` control sequence specifies the file that contains graphics data, plus the height and width of the area the plot is supposed to fill. The file will automatically be scaled and drawn. The `\special` command is actually generated by library macros that also build the enclosing vbox. This command, for instance, drew a square, column-width Tektronix plot in the last *TUGBOAT*:

```
\graph{\file={bpr}
  \width=\columnwidth
  \height=\columnwidth}
```

The `\graph` macro has other, optional parameters that can specify a border and a surrounding rule. We presently support three graphics sources: Tektronix and Calcomp plots, and Macintosh MacPaint pictures.

I had the good fortune to already have a printer and fonts available when I started this project; these had been previously purchased from Talaris Systems, Inc., for use with a lesser formatter. Because of this, I did not have to do any conversion of `PXL` files. Unfortunately, I do not own the fonts and therefore cannot distribute them.

Finally, I would like to thank Pierre MacKay for providing me with the latest sources and for encouraging me to become a site coordinator for this implementation of TeX.

Jim Fox
University of Washington

Data General Site Report

We have successfully ported TeX 1.5c. Of course, it was absolutely no problem. The previous change file was sufficient. We use the "c" suffix to indicate that it uses the CM family of fonts. (See the next item.) I have not had enough time to create the program for using non-virTeX's.

We have upgraded our METAFONT port to 0.999999. I have spent a day or two making the `.GF` files. I am now testing these in preparation for a new release tape. It took about a day of CPU time on the MV10k to make the complete CM family at magsteps zero, half, one, and two.

As soon as I can complete the program for non-virTeX's and finish testing the new fonts, I will start sending the new distribution tapes out.

I have had a lot of requests for the utility items. These are available on MS-DOS diskette or several reasonable tape formats.

Bart Childs
Texas A & M University

Macintosh Site Report

TEX for the Apple Macintosh is available now, in a pre-release package for experienced TEX users. MACTEX runs on Macintosh XL, Macintosh 512, and Macintosh Plus computers; one double-sided or two single-sided floppy disk drives are required, with a hard disk recommended for large documents. MACTEX combines TEX82 version 2.0 and the new CM Computer Modern fonts with an integrated text editor, screen display viewer, and print drivers for the Apple ImageWriter and Apple LaserWriter.

The standard memory allocation (`mem` = 30000 words) is available in a 512KB Macintosh (or Switcher partition). Additional memory is automatically used to extend `mem` to 64000 words, allow more fonts, improve performance by reducing segment swapping, and to permit text editing to continue during TEX processing. In a 1 MB system, typical page processing time is 10-20 seconds.

The text editor supports very large files, multiple active windows, and the standard Macintosh selection, cut, and paste operations.

The screen display viewer accurately displays typeset documents, with random access to any page and a viewing magnification instantly changeable to any value from 100 through 5000.

Due to the high interest expressed, we are making the MACTEX package available for pre-release distribution (*translation: we'd like to get some of you out of our hair so we can finish it*). This pre-release package contains everything described above, but has only sparse documentation, several loose ends, and some missing features. However: it's truly a Macintosh program, so experienced TEX users will have no difficulty without a manual; and it includes automatically, at no extra charge, a copy of the published release (version 1.0) when available.

The published version 1.0 will have, among other niceties, the ability to include MacPaint and MacDraw pictures in TEX documents.

To obtain a copy of the pre-release package, contact Brenda Cavallaro, Addison-Wesley (EMSD), Reading, Massachusetts, 01867, or call her at (617) 944-6795.

Barry Smith
Kellerman & Smith

UNIX Site Report

Since October, I have been the only resident site coordinator for UNIX TEX, owing to Richard Furuta's departure for the University of Maryland. This will in part explain some of the delays in delivery which have intervened at various times, and arbitrary hardware failures or shortages of magnetic tape account for the rest. The backlog has been unmistakable evidence of the increasing interest in UNIX TEX, and I find myself wondering what the expected increase in the range of target machines will bring. We are still unable to provide anything guaranteed to work on System V machines, but the interest is now so widespread in both the System V and Xenix worlds, that we expect to hear of a free public-domain port to one or the other system before the end of the year. Richard Furuta and I still collaborate over the electronic mail network, and it is possible that our position on opposite sides of the country is actually an advantage in our search for new contributions.

The question has sometimes been asked, particularly at times when the backlog piled up, why we maintain a separate distribution at all. The reason lies in the nature of the UNIX TEX distribution, which is very much shaped by the particular character of the UNIX system itself. What we offer on our tapes is not just a collection of change files but something as close to a sort of turnkey system as we can manage. It is not quite possible to put a UNIX TEX distribution tape on the drive, copy it, type `make tex` and go home, but we have attempted to arrange files in such a way that we could actually provide a `Makefile` which would do that, if we really thought that any rational systems programmer would want to try it. Moreover, the UNIX TEX distribution has attracted to itself a rich variety of supporting programs, about half of which are quite specific to the UNIX system, and it is constantly attracting more. We are trying to offer much more than a set of change files which will bring up the various programs directly related to TEX and METAFONT. We are trying to offer an entire TEX-users environment, working in the UNIX system. In recent months we have been able to make that environment more comfortable and less restrictive by separating the 4.1 BSD and the 4.2/4.3 BSD distributions. As a result, we do not ask for a BSD source license any longer except from those few sites that still use 4.1 BSD. This also removes any restrictions about recopying and redistribution, though we do insist that any redistributed copy be complete, and include all the files that were sent out with the original tape.

The past six months have seen major changes in just about every directory on the tape. TEX is now offered at version 2.0, LATEX is offered at version 2.09 (consistent with the manual published by Addison-Wesley), and METAFONT is offered at version 1.0. This last item is the most significant change on this occasion. Paul Richards of the University of Illinois has made a complete METAFONT system available in the `./mf84` directory, together with all the significant METAFONT-ware. The new version covers a number of different target systems, and appropriate `Makefiles` are created through an interactive `configure` script. Here, on one of the local 4.3 BSD UNIX machines, METAFONT came up absolutely smoothly, with no difficulties at all, and passed the `trap` test with complete success.

The approach to compilation on SUNs now assumes that the SUN assembler (the last stage of a `pc` compilation) is now capable of dealing with a unitary METAFONT (or TEX) file in something under a week of elapsed time. For those who are still running the old "whirling dervish" assembler, there is a `split_source` script to allow for compilation in four chunks. The problem with the old assembler has been identified by one of our correspondents. It includes "optimizing code" whose execution time increases as the cube of the number of statements in the source file. At the time of writing there are still some problems with SUN3 software. The old `undump` program no longer works, owing to a change in the format of both `core` and `a.out` files, and on some versions of the software there is an unexpected and, we trust, unintended limitation on the array bounds in SUN Pascal. We expect that both these problems will soon be resolved, and we have reports of sites which have successfully compiled `initex` and `virtex` using SUN3 software, which probably indicates that the array bounds problem is already corrected.

Paul Richards's `configure` script includes options for the Pyramid which he has validated. We have not yet had the opportunity to try them out at the University of Washington. We are also expecting change files for Pyramid compilations of TEX in the immediate future.

TEX 2.0 is functionally identical with the most recent release of TEX 1.5. The only change in the Pascal code comes in the addition of a couple of lines to clean up terminal interaction at one point in the program. The real significance of the new version is that it implies the use of `cm` fonts in place of the `am` fonts (which were modified versions of a yet earlier set of `cm` fonts). There is no impediment at all to preloading the old `plain.tex` into `virtex` 2.0, and we have therefore gone ahead with the distribution of TEX 2.0 on the tapes now being written. For about six months, we plan to continue with `am` fonts as the basic option, and `cm` as the alternate. The new versions of `plain.tex` and `webmac.tex` are provided under the names `cm.plain.tex` and `cm.webmac.tex`. As soon as all the supplementary LATEX fonts are available in new METAFONT format, we will switch over totally to the new fonts, but we will then allow a transition period during which `am.plain.tex` and `am.webmac.tex` remain on the tape as alternatives.

This brings us to a serious consideration of fonts and their effect on the sheer size of the distribution. The only thing that has made it possible to work out a relatively painless transition of this sort, with both varieties of fonts available simultaneously, is the timely release of Tomas Rokicki's PK format and its associated utility programs. As font styles and sizes proliferate, the storage requirement for the loosely packed PXL format becomes excessive. In the next few years we can expect that PK format will replace PXL format throughout the TEX community. (This change presents a new argument for WEB-coded output drivers, incidentally, since the essential code to unpack PK format can be patched into a WEB-coded driver directly out of Rokicki's `pktopx` program.)

On the distribution tape, we have begun by packing the entire list of 300 dpi fonts, leaving only the `*.1500pxl` fonts available in the loosely packed version. The 200/240 dpi fonts are all in `pxl` format still, but that is solely because they also serve as the working font library at this site. As we move into the full conversion to CM fonts, we hope to find enough space to offer both the CM and the AM fonts on the tape for a few months, but eventually we must be ready to drop the AM fonts altogether. We would urge all sites receiving UNIX TEX to keep this in mind, and to start preparing for the change now. The transition will be relatively painless if AM versions of favorite macro files are prepared in advance, and stored away for the arrival of "CM-day."

Among the last productions from the old version of METAFONT-in-SAIL program is the large collection of Cyrillic and

Special Symbol fonts which the American Mathematical Society has generously offered for free, unlicensed distribution. These are the fonts described and illustrated in *TUGBOAT* 6, no. 3: 124–128, and they are included on the tape in a separate directory `./amsfonts`, together with the essential macros used to call them into text.

In order to get all this new material packed down to fit onto a single reel of tape, we have had to resort to compression on an increasing number of text files. On the latest tapes, all the files in `*/doc` are compressed, using a very efficient program collected from `net.sources` and forwarded to us by Paul Richards. We have put this onto the tape in a separate `./compress` directory, so that those who have no immediate access to the net can `uncompress` these files.

Work on new drivers is continuing throughout the world. We have news that the long-awaited LN03 driver may soon appear, and a recent communication from Helsinki offers a driver for the HP Laserjet$^+$. For those who set their sights on something beyond the limits of dry toner resolutions, I recommend keeping a lookout for the announcement of a new 2400 dpi laser-diode photo-typesetter that will become available at well below $10,000, perhaps even as low as $5,000. That ought to trigger the next drop in the price of dry toner print engines to something like $1,000 apiece.

I have not even attempted to list all the names of individual contributors to the UNIX TeX distribution over the past six months, but that in no way diminishes my appreciation of their assistance. I should like instead to point out how the continuing growth of the UNIX TeX collection of free software justifies Richard Stallman's predictions at the time when the Free Software Foundation was being organized, that a very large number of very superior programmers will be quite ready to contribute their efforts to the enhancement of the entire programming environment. Some of the policies of the UNIX TeX distribution have been revised with the specific aim of bringing them more closely into line with the policies of the Free Software Foundation. The Free Software Foundation has chosen TeX as the appropriate vehicle for documentation of its programs, and I have recently had the pleasure of helping to set the GNU Emacs manual into type. We have by these actions started on a program of cooperation and resource sharing by which we hope to accelerate the collection and development of software tools which will be made freely available throughout the world to all competent users.

Pierre MacKay
University of Washington

VAX/VMS Site Report

"By the time you read this" — a handy phrase, attributed to D. Fuchs — we should have completed a VAX/VMS package containing the latest versions of TeX82 (2.0), LaTeX (2.09), METAFONT(1.0), the new Computer Modern fonts (1.0), and all of the related TeXware and METAFONTware programs.

The package will also include additional software on an unsupported basis: Andrew Trevorrow's DVItoVDU preview display driver described elsewhere in this issue, and public domain drivers for the Versatec and LN03 printers. (Note that Kellerman and Smith also offer commercially supported Versatec and LN03 drivers as separately priced items.)

The above package includes executable images of all programs (VMS 4.2 or later), all sources and build command files, a copy of the TeXbook, and 150 pages of VAX/VMS specific documentation including turnkey installation procedures. The package is supplied in BACKUP format on a 1600 bpi, 2400 foot magnetic tape, and costs $200.00 (U.S.) including shipping within the U.S. and Canada. Add $50.00 (U.S.) for air freight shipment to other countries.

Note that this package no longer includes the Almost Computer Modern (AM) fonts, and that it requires VMS 4.2 or later (or at least the recent VMS Pascal library). We will continue to make the previous TeX 1.3 VMS release available on request, for the same distribution fee.

Barry Smith
Kellerman & Smith

LaTeX News

Starting with this issue, I will try keep users abreast of the latest LaTeX news. I expect this news to be dull; LaTeX was designed to be dependable, not exciting. No major bugs have been discovered and no noticeable enhancements are planned.

The first printing of the LaTeX manual sold out quickly at many bookstores. A large number of copies from that printing are apparently on a boat en route to Timbuktu, so Addison-Wesley rushed out a small second printing. No corrections were made to that printing.

Two new document-style options have been added: `bezier` for drawing curves and `ifthen` with conditional evaluation and looping commands. A document style that will format text for the ACM "transactions" journals is in preparation, and I will be negotiating with the ACM to allow authors to submit either camera-ready copy or LaTeX input files.

I suspect that many sites have installed LaTeX without installing the appropriate human system for maintaining it. There should be a site coordinator who is responsible for installing LaTeX (with any necessary site-specific changes), creating and maintaining the *Local Guide*, fielding questions from users, and obtaining the latest versions of LaTeX files.

Leslie Lamport
Digital Equipment Corporation

Multilingual TeX Update

This note updates the extension to TeX that allows for multilingual hyphenation reported in *TUGBOAT* 6, no. 2 (July 1985): 57–58. A key feature of the extension is that it accommodates *standard TeX fonts*, including words with accented letters. For details of the features the reader should refer to the *TUGBOAT* report. The changes and retractions are as follows:

TeX is now truly multilingual. The restriction on the `trie_op` size has been removed. It is now possible to accommodate up to 65000 languages – although TeX currently has a consistency check that arbitrarily restricts it to 100.

Contrary to what was reported in the previous article, TeX cannot change hyphenation rules on a word by word basis. It is restricted to the language in force at the end of a paragraph. The reason for this is that the value of the `language` parameter is not carried along with the character in the same way as the font information. Most applications should be satisfied with paragraph by paragraph hyphenation. For those that are not, an extension involving an increase in the size of a `char` node is possible.

Michael J. Ferguson
INRS-Télécommunications

TeX Notes

Automatic Page Sizing

Picture a 50-page manual with 50 topics, each topic needing about a page. How could I automatically get a page-filling combination of margin size, magnification, openup, etc., so that *each designated topic will fill its page*? I'm picturing a camera's "program" mode, where the designer decided in advance how to balance the aperture-size/shutter-speed so that the light, whatever its quantity, would always "fill" the film. Just as each photo has different exposure parameters, each page of the manual could be slightly different. The only form of automatic *page* control for TeX seems to be through inter-word spaces and hyphenations, but not through line length, margins, and interline spaces. Of course, page-filling, whether through a program or through trial and error, can work only within certain quantity limits. As with the camera, it would be good (but not necessary) if there were a means of over-riding the automatically-chosen parameters, so that the user could, for instance, put priority on either magnification or margin. It would be nice if appropriate horizontal and vertical offsets were included in the program, and if the user could direct that the inner margins be so much larger or smaller than the outer.

I look forward to reading—and seeing—the reply.

Alan Auerbach
Wilfrid Laurier University

Editor's reply: On page 59 of *The TeXbook*, it is decreed "You cannot apply two different magnifications to the same document." So `\magnification` is not a possible variable.

It should be possible to simulate the effect of magnification by defining suitable typesize groups in the manner of `\tenpoint`, etc. (*The TeXbook*, page 414), perhaps naming them `\tenpointmaghalf`, `\tenpointmagone`, etc., for larger sizes where specific magnified fonts have been loaded explicitly, as `\font\tenmaghalfrm=\cmr10 scaled \magstephalf`, etc. (The user should be careful to set appropriate values for baselineskips and the dimensions of strutboxes.)

Modifications to margin width, line length, and interline spacing can all be applied in a relatively straightforward manner to the text, while manipulation of top, bottom and side margins is probably most easily implemented through parameters in the output routine. The biggest challenge seems to be in deciding how much adjustment is needed to each page (after test-setting it using "standard" values), and recycling the data using the revised parameter values.

However, just because implementing a format of this sort might be possible doesn't mean that it's always a good idea. (After all, TeX aims to be a tool which will permit typographers to create documents of the highest quality when judged according to traditional standards.) Noticeable variations in spacing, size and character style can distract from the main business of a document, which is to deliver a message clearly without making the reader consciously aware of the document's appearance. If the extreme cases diverge too greatly from the norm, then probably some different kind of solution is wanted.

Barbara Beeton

Where to Find TeX File Descriptions

The TUG office frequently receives calls asking in what issues of *TUGBOAT* various file descriptions can be found. Here is a list of the pertinent references; information applicable only to TeX78 has been omitted.

`DVI` Device Independent file, produced by TeX, read by output device interfaces.
TUGBOAT 3, no. 2: 14–19.

`GF` Generic Font file, produced by METAFONT, read by output device interfaces.
TUGBOAT 6, no. 1: 8–11.

`PK` Packed font raster file, produced by GFtoPK, read by output device interfaces.
TUGBOAT 6, no. 3: 115–120.

`PXL` font raster file, produced by METAFONT78, read by output device interfaces (obsolete, but still widely used),
TUGBOAT 2, no. 3: 8–12.

`TFM` TeX Font Metric file, produced by METAFONT, read by TeX and output device interfaces.
TUGBOAT 2, no. 1: 12–16.

Much of the information in the TeX Users Group articles has been extracted from `WEB` source files, so the programs listed below are also useful references.

`DVI` TeX82, §583
`GF` METAFONT84
`PK` PKtoPX
`TFM` TFtoPL and PLtoTF

THE PLAIN TRUTH: DISPLAYLINES, IALIGN

Barbara Beeton
American Mathematical Society

This column continues the attempt, begun in the last issue, to illustrate the reasons for various changes to `plain.tex`. This issue's installment covers various changes made to macros controlling alignment.

Displayed Lines of Equations

The `\displaylines` macro lets you display any number of formulas in any way you want, without any alignment between formulas. Its original definition looked like this:

```
    \def\displaylines#1{\displ@y
      \halign{\hbox to\displaywidth
        {$\hfil\displaystyle##\hfil$}\crcr
         #1\crcr}}
```

The first change never actually got into print in an errata list, being posted and rescinded almost immediately, but it may have been included in some distributions of version 1.1, or picked up via the Arpanet by unsuspecting users. This change consisted in putting braces around the aligned argument `##`.

These braces were added to be consistent with similar syntax in the definitions of `\eqalign`, `\eqalignno` and `\leqalignno`, but there is a significant difference — the `eqalign` macros were designed to produce only certain explicit structures, and `\displaylines` is provided to handle the nonstandard cases, where pieces may have to be moved around by hand. The following use of `\displaylines` is common:

```
    \displaylines
       {\rlap{($\ast$)}\hfill a+b=c\hfill}
```

Without braces — `\displaystyle##`:

$(*)\qquad\qquad\qquad a+b=c$

With braces — `\displaystyle{##}`:

$$\rlap{$(*)$}a+b=c$$

The `\hfill` instructions, intended to overpower occurrences of `\hfil` in the definition, lose their effect within braces.

Another bug in `\displaylines` was flushed out with the aid of this expression:

```
    (m)\underbrace{x+y}(n)=0
```

This behaves nicely in a simple display, `$$...$$`:

$$(m)\underbrace{x+y}(n)=0$$

But with the "uncorrected" `plain`, the result of `\displaylines{...}` was unexpected:

$$(m)\,{}_{\underbrace{x+y}}(n)=0$$

(Look closely at the baseline.) The explanation hinges on the complicated expansion of `\everycr` in the expansion of `\displ@y` (it appears in *The TeXbook* on page 362 and won't be repeated here); `\everycr` needs to be reset, and that was the nature of the fix to `plain`:

```
    \def\@lign{\tabskip=0pt \everycr{}}
    \def\displaylines#1{\displ@y
      \halign{\hbox to\displaywidth
        {$\@lign
          \hfil\displaystyle##\hfil$}\crcr
         #1\crcr}}
```

`\eqalignno` and `\leqalignno` were changed in a similar manner, by inserting `\@lign` before every instance of `\displaystyle` in their definitions. The `\tabskip=0pt` in `\@lign` locally resets `\tabskip=\centering` in `\*eqalignno` in case an alignment occurs in the argument.

Initialized Alignment

The `\ialign` macro provides an `\halign` for which `\tabskip` is initially zero. Its need for initialization similar to `\displaylines` had been discovered earlier. The original definition looked like this:

```
\def\ialign{\tabskip=0pt \halign}
```

The correction was:

```
\def\ialign
  {\everycr{}\tabskip=0pt \halign}
```

`\ialign` occurs internally in many `plain` macros, including ones dealing with alignment using tabs, accent placement, placement of arrows or braces above or below expressions, construction of “composite” characters from separate symbols (e.g. $\cong$ from $\sim$ and $=$), matrices, and displays aligned on equal signs.

To be continued...

Multiple Changefiles in WEB

Wolfgang Appelt
Karin Horn
Gesellschaft für Mathematik und Datenverarbeitung mbH

TANGLE and WEAVE usually read one *webfile* and one *changefile* to produce the desired Pascal source file or the corresponding TEX input file. There are, however, situations when it would be useful if TANGLE and WEAVE could read several *changefiles* simultaneously to create their output files.

Imagine, for example, a TEX site where TEX is running on several computers with different operating systems (say $s_1, \ldots, s_n$) and where different output devices $(o_1, \ldots, o_m)$ are supported (as is the case at GMD). If a *changefile* for the DVItype program is written to create a driver for the output device o_j running on the system s_i, the *changefile* will usually contain a set of changes, say $\mathcal{A}_i$, which only concerns the operating system but not the output device. A second set of changes, $\mathcal{B}_j$, may only concern the output device and a third one, $\mathcal{C}_{ij}$, may concern both the operating system and the output device. (This set may be empty on many systems.) There might even be a set of further changes, $\mathcal{D}$, to support some \special features, e.g. for graphics. In other words, a *changefile* for a specific output device on a specific system can be regarded as the union $\mathcal{A}_i \cup \mathcal{B}_j \cup \mathcal{C}_{ij} \cup \mathcal{D}$ where each of these subsets is logically independent from the others.

Basically, there are two possible ways to store the *changefiles*:

(1) For each combination of an operating system and an output device there exists one complete *changefile*. Not only is space wasted in this way; an even greater disadvantage of this method is that whenever a modification is necessary (maybe because a bug was found or because a new system release was installed), the same modification would have to be applied to several *changefiles*.

(2) All the different sets of changes $\mathcal{A}_i$, $\mathcal{B}_j$, $\mathcal{C}_{ij}$ and $\mathcal{D}$ are kept in separate files. Only if a specific driver program has to be created are the required files merged to create a valid *changefile*. Merging these files, however, might not be a trivial task, since a simple concatenation of $\mathcal{A}_i$, $\mathcal{B}_j$, $\mathcal{C}_{ij}$ and $\mathcal{D}$ is usually *not* sufficient.

A better solution which avoids these problems would be a version of TANGLE and WEAVE that can process more than one *changefile*. We have therefore written two programs which we call KNIT and TWIST which implement that feature. (The sum of the two programs might be called the PATCHWORK system).

The philosophy for handling several *changefiles* is as follows: Assume we have *n changefiles*, called *change_1* ... *change_N* and, furthermore, assume that a line of text which appears between @x and @y in any *change_i* does not appear in any other *change_j*, i.e. all changes appearing in the *changefiles* concern *distinct* parts of the *webfile*. In this case the output files of KNIT and TWIST are identical* to those obtained by TANGLE and WEAVE with a *changefile* which is created by merging *change_1* ... *change_N* properly together.

If, however, two (or more) *changefiles* want to change the same piece of text within the *webfile* only the modifications by the *changefile* which claimed its right first will take effect; the others are ignored. In other words, in such a case the numbering of the *changefiles* (*change_1* is read before *change_2*, etc.) is important. KNIT and TWIST will give a warning if two *changefiles* want to change the same text of a *webfile* since this is probably an error. (Nevertheless, there can be situations where one may deliberately construct "conflicting" *changefiles*.)

* To be precise, there is a *slight* difference: Changed modules are marked with the number of the *changefile* which caused the modification and not just with an asterisk as WEAVE does.

The KNIT and TWIST programs were created by writing two *changefiles*, namely **knit.chg** and twist.chg. TANGLE'ing **tangle.web** with **knit.chg** will result in **knit.pas**, the Pascal source program for KNIT, and correspondingly TANGLE'ing **weave.web** with **twist.chg** will give you **twist.pas**. Setting up the KNIT and TWIST processors is therefore similar to bootstrapping the WEB system.

Error Detecting Changes to TANGLE

R. M. Damerell
Royal Holloway & Bedford College

This article describes a proposal to improve the diagnostics of TANGLE by making it detect certain types of errors. No corresponding change is proposed for WEAVE, on the theory that users are unlikely to want to weave a program until the worst of the bugs have been fixed. I would be much obliged if members of the TeX community could say whether these changes are desirable. If the response is favourable, then I propose to make copies of the modified TANGLE freely available. The changes described in this article apply to version 2.5 of TANGLE, as supplied by Stanford.

Change 1: Missing Start of Module

By far the most frequent and troublesome error that I make when writing WEB programs is that of omitting the string @␣ that should separate two successive modules. This error usually damages the PASCAL output. Consider the following fragment, copied from TANGLE:

```
    @<Globals in the outer block@>=
    @!history:spotless..fatal_message;

    @ @<Set initial values@>=
    history:=spotless;
```

Now suppose the user omits the @␣ between these modules. Then the whole replacement text of the second module will be inserted into the first. If this error is not detected, then probably the user will try to compile the damaged output. The likeliest result is that the compiler will become confused and print lots of error reports that bear no clear relation to the cause of the trouble. There seems to be no obvious way to detect this type of error every time, but the changes here proposed will detect it with a fairly high probability.

```
@<Globals in the outer block@>=
  try_loc: integer;

@ @<Was an '@@' missed here?@>=
try_loc := loc;
while (buffer[try_loc] = " " )
  and (try_loc < limit) do incr(try_loc);
if    (buffer[try_loc] = "+" )
  and (try_loc < limit) then incr(try_loc);
while (buffer[try_loc] = " " )
  and (try_loc < limit) do incr(try_loc);
if    (buffer[try_loc] = "=" ) then
  begin err_print
    ('! Nested named modules. Missing @?');
      @.Nested named modules@>
  @<Show list checkpoint@>;
  end
```

Figure 1. *Code to Detect Missing @␣. The procedure* scan_repl *is changed to execute this code when it has just read a module name and* loc *is pointing to the character that follows the* @> *at the end of the name.*

First, I have changed TANGLE so that it enforces an extra rule of syntax: if the replacement text of one module includes a call of another module, then that call may not be immediately followed (on the same line) by = or +=. This rule is not really valid for WEB programs: somebody might want to say:

```
    if @<some expression@> = 0 then ...
```

but this construction does seem unlikely in PASCAL. Also the test can be bypassed by putting the ' = 0' onto the next line.

Figure 1 shows the code that performs the test. TANGLE has just read a module name, say NNN, within the replacement text on another module, say MMM. Then loc should be pointing to the next character after the @> that ends the module name NNN. TANGLE must then examine the next few characters in the file without losing its position.

That module must be called by inserting

```
@<Was an '@@' missed here?@>;
```

immediately before the call of `app_repl` which follows the label `module_name` in procedure `scan_repl`.

Change 2: Showing a checkpoint

The next change is suggested by the fact that several errors, which are currently detected by `TANGLE`, nearly always occur as symptoms of a missing `@␣`. A typical example is "Identifier conflict." Here is how this happens. Suppose the programmer omits an `@␣` between two modules. Then `TANGLE` will read the TeX part of the next module as if it were PASCAL. When it reads a sentence like "If this fails, then ...," the "If" conflicts with PASCAL's `if`. The obvious change here is to alter the error message to warn the user of this possibility. Also, I have inserted a call of

```
@<Show last checkpoint@>;
```

immediately after the code that generates the message. The messages "`@d/@f/@p ignored in PASCAL text`" also seem to occur frequently as symptoms of a missing `@␣`, and I have treated them in the same way.

When `TANGLE` thinks it has found a missing `@␣`, we would like to tell the user where it ought to have been. The best I can do is to indicate a range of line numbers. The actual error is that two consecutive modules, say `MMM` and `NNN`, have been run together; the formal error is an illegal construction within the PASCAL code of module `MMM`. So `TANGLE` will say where it thinks the PASCAL code of that module started.

Change 3: Semicolon-Else

Another frequent error is that of writing a program containing the sequence ‘ `; else` ’, which PASCAL does not allow. This is easy to do, as the semicolon and the `else` often come from different modules. Although there is absolutely no reason why you cannot say ‘ `; else` ’ in `WEB` programs, it still seems desirable that `TANGLE` should detect this, as (on our machine, at any rate) `TANGLE` runs between 5 and 10 times as fast as the PASCAL compiler. The test applied here is very crude; it will not recognise `ELSE` or `Else` or the output from ‘ `el@& se` ’.

The module to detect the change, listed in Figure 3, gets called by inserting

```
@<Semi else test@>;
```

```
@p procedure scan_module;
  label continue, done, exit;
  var p: name_pointer;
  {module name for the current module}
  begin incr(module_count);
    @<Scan the \(definition part...@>;
    last_line := line;                          — insert
    was_changing := changing;                   — insert
    @<Scan the \PASCAL\ part...@>;
    exit:
    last_line := 0;                             — insert
  end;

    :

@<Globals in the outer block@>=
last_line: integer;
was_changing: boolean;

@ @<Show last checkpoint@>=
if last_line = 0 then
  print_ln(' (not in PASCAL)')
else
  begin
  print_nl('PASCAL part of module',
    ' began at line ' , last_line);
  if was_changing then
    print(' in change file');
  print_ln(' ');
  end

@<Set initial values@>=
last_line := 0; {where PASCAL started}
was_changing := false;
```

Figure 2. *Code to Show Checkpoint. The lines added to the* `scan_module` *procedure track the starting line of the latest piece of PASCAL code. Notice the checkpoint is reset at the beginning of the program and after the end of the PASCAL part of each module. The module* `@<Show last checkpoint@>` *displays the saved checkpoint; references to this module are added following the error reporting code for errors that are often caused by a missing start of module.*

immediately after the label `reswitch` in procedure `send_the_output`.

Change 4: Error reports to a file

Another change that I have found very useful is to make TANGLE write its error reports onto a file as well as the terminal. With a split screen editor, one can work through the error and source files in parallel. This is much easier than writing the errors on paper as they appear. Essentially, you say:

```
    @d print(#)== begin write(term_out,#);
       write(errorfile,#); end
```

but several refinements are needed to prevent routine messages from getting into the error file. There seems to be no point in giving details here as they are long and messy and system dependent.

Conclusion

In designing these changes, I have tried to ensure that the new version of TANGLE will be compatible with the old. So when TANGLE thinks it has found an error, it merely prints an error report without making any attempt to correct the supposed error. The missing @␣ test is fairly effective: there are only about 10 places in WEAVE.WEB where you can omit an @␣ without it being detected. The semicolon-else test is also effective, provided that the user always writes else in lower case. The modified TANGLE seems to run about 2% slower than the old version; but users will save more machine time by not trying to compile bad PASCAL programs. I believe that these changes will significantly reduce the time that programmers have to spend in removing trivial errors from WEB programs.

```
@<Globals in the outer block@>=
point_else: name_pointer;
semi_last: boolean; {output was semicolon}
@ The first step is to put an '|else|' into the hash
table,
to be used in later comparisons.
@<Initialize the input system@>=
  buffer[0] := "e"; buffer [1] := "l";
  buffer[2] := "s"; buffer [3] := "e";
  id_first := 0;
  id_loc :=4;
  point_else := id_lookup(normal);
  buffer[0] := " ";
  semi_last := false;

@ Then |send_the_output| must test
for ' ; else ', ignoring intervening
comments or white space.

@<Semi else test@>=
  if   (cur_char = begin_comment)
    or (cur_char = join)
    or (cur_char = module_number)
    or (cur_char = 0)
    or (cur_char = force_line)
    or (brace_level > 0)
  then do_nothing
  else if cur_char = ";" then
    semi_last := true
  else if semi_last
   and (cur_char = identifier)
   and (cur_val = point_else)
    then
      begin err_print
       ('! semicolon-ELSE found');
         @.semicolon-ELSE found@>
      semi_last := false;
      end
  else semi_last := false;
```

Figure 3. *Code to check for semicolon - else combination. A reference to the module* `@<Semi else test@>` *is added to the main loop of* `send_the_output`, *following the* `reswitch` *label, to check the program as it is expanded and output.*

DVItoVDU: A TeX Page Previewer

Andrew Trevorrow
University of Adelaide

DVItoVDU is an interactive program that allows the user to view pages from a TeX82 DVI file on a variety of commonly available visual display units (VDUs). It runs under VAX/VMS and is written in Modula-2 from the University of Hamburg.

The software is in the public domain and available on the VAX/VMS distribution tape. Most of this article is based on material from the *DVItoVDU User Guide* and the *DVItoVDU System Guide*. The TeX source files for these two documents are also on the distribution tape.

The version described below is numbered 1.5 (October 1985).

Conception

TeX usage began at Adelaide University early in 1984 and the clamour for a previewing program started soon after. Although there are excellent, rational reasons for such a tool in a TeX system, our users had somewhat more pragmatic concerns: people had to come to the Computing Centre to collect their laser printer output; there was a charge of 10 cents per page! (This has since been dropped.)

Another concern was the high level of paper wastage, particularly in those early days when most users, including myself, were learning about TeX and all its intricacies. While the need for a previewer was obvious, I could see a number of difficulties in writing such a program.

Around this time I happened to stumble upon Hamburg's Modula-2 system. After a few weeks of pleasant experimentation with this new language, I eventually realized that Modula-2's procedure variables provided an elegant, high-level solution to one of the key design problems for the particular previewer I had in mind; that is, the need to efficiently drive a variety of terminals from the one program. Up until then I'd been reluctant to seriously consider starting the project. The thought of having to resort to VAX Pascal wizardry or MACRO magic was just too depressing.

Why not `WEB`? Quite apart from the above design problem, it was never really a serious contender. At the risk of being sacrilegious I must confess to having some reservations about the `WEB` system. I believe its benefits are outweighed by the disadvantages of using a cryptic language in which `WEB`, TeX *and* Pascal errors are all possible.

Modula-2 in fact matches `WEB` in its facilities for creating highly modular programs. In addition, system generation is much faster because Modula-2 allows separate compilation. It is usually possible to make a change to an implementation module, compile and link, and have a new EXE file in seconds. Compare this with the many minutes normally needed to change a `WEB` module, run `TANGLE`, then Pascal and the linker (not to mention `WEAVE`, TeX, etc., if you want your documentation up-to-date).

A modern language such as Modula-2 and a good screen editor are sufficient tools, I believe, to create well-structured software with good-quality, internal commentary. Having access to TeX or some other typesetting system to create accompanying documentation is an added luxury.

I was also keen to write a fairly large program in Modula-2 to see how it compared with Pascal. I must say I was pleasantly surprised by the utility of Modula-2 and the reliability of Hamburg's compiler.

Design Considerations

The main design goal was to have just the one preview program able to work efficiently on various types of terminals used throughout the campus. Since many TeX users do not have access to a high-resolution graphic VDU, the program also had to produce useful displays on a simple ANSI terminal, such as a VT100. A number of other capabilities were considered essential:

- Absolute page selection using either the natural DVI page order or the TEX page counters.
- Relative page selection by requesting the next page in either direction.
- "Pan and zoom" (the ability to view any region of a selected page, and at any desired scale);
- Error detection in the form of explicit warnings about such problems as a page off the paper or the use of a non-existent font size.

A preliminary version of DVItoVDU that met most of these goals was released in September 1984. A number of substantial changes have been made since then, mainly to improve efficiency and to provide a more flexible user interface.

Running DVItoVDU

The information in this section is a condensation of the *DVItoVDU User Guide.*

We'll assume you've just run `foo.tex` through TEX to create `foo.dvi`. To look at the pages in this DVI file you simply type '`dvitovdu foo`'. Some command options may be necessary if DVItoVDU is to work properly. In particular, the `/vdu` qualifier must correctly describe the type of terminal you are using.

The DVItoVDU command can be followed by a number of qualifiers, where each is assigned a value:

`/vdu`	type of terminal
`/resolution`	pixels per inch
`/xsize`	paper width
`/ysize`	paper height
`/magnification`	override magnification
`/font_directory`	master font directories
`/dummy_font`	used if font not found
`/help_file`	used by `?` command

The last three are really for system wizards; their default values should be set up so that most users need never worry about changing them. (The DVItoVDU command is installed in the system DCL tables using a command language definition file supplied with the software. This CLD file can be modified to specify default qualifier values suitable for your site.) Let's look at all the qualifiers in more detail:

VDU Type The `/vdu=`*string* qualifier is used to tell DVItoVDU what type of VDU you are using. *string* is a string of characters terminated by a space or the start of the next qualifier. Most sites might set up '`/vdu=ANSI`' as the default. If `ANSI` does not describe your VDU, you need to override the default value. For example, if you're using a VISUAL 550 terminal, type '`dvitovdu/vdu=vis550 foo`'.

The current version of DVItoVDU will accept the following `/vdu` values:

`AED483`	AED with 512 by 483 screen
`AED512`	AED with 512 by 512 screen
`ANSI`	any ANSI compatible VDU
`REGIS`	any ReGIS compatible VDU
`VIS500`	VISUAL 500
`VIS550`	VISUAL 550
`VT100132`	VT100 in 132 column mode
`VT640`	VT100 with Retro-Graphics

`VT100` and `VT220` are synonyms for `ANSI`. `GIGI`, `VK100`, `VT125` and `VT240` are synonyms for `REGIS`.

Printer Resolution The `/resolution=`*i* command tells DVItoVDU the resolution of the device that will be used to print your document. DVItoVDU treats the imaginary sheet of paper on which a DVI page will appear as a two-dimensional array of tiny dots known as "paper pixels." *i* is a positive integer that defines the number of paper pixels per inch, horizontally *and* vertically. We have an Imagen IMPRINT-10 laser printer, so our default `/resolution` value is 240.

Printer Page Size The `/xsize=`*dimen* and `/ysize=`*dimen* qualifiers qualifiers define the dimensions of the paper upon which your document will be printed. `/xsize` defines the width and `/ysize` the height. Every time you select a page, DVItoVDU will use these paper dimensions to check that the page edges fall within the paper edges. *dimen* is a positive integer or real number followed by a two-letter unit: `in`, `cm`, `mm`, `pc`, `pt` or `px`. Most of these should be familiar from TEX. DVItoVDU provides an additional unit, `px`, for paper pixels. (These two-letter sequences are the same as the commands used to change the units of dimensions; more about all the commands on page 28.) Our laser printer uses A4 paper by default; i.e., `/xsize=8.3in` and `/ysize=11.7in`.

```
Total pages=n   DVI page=0   TeX page=[0]   Next=>   Terse
Window at (h,v) wwd by wht   Page at (minh,minv) pwd by pht   IN
status
Command:
```

Figure 1.
The Initial Dialogue Region.

Explanation of entries:

`Total pages=`*n*	The total number of pages in the DVI file.
`DVI page=0 TeX page=[0]`	The current page number and its corresponding TEX page counters.
`Next=>`	The direction the `N` command will advance through pages (initial setting is forward).
`Terse`	The current display mode — one of `Terse`, `Box`, and `Full` (initially terse).
`Window at (`*h*`,`*v*`)` *wwd* `by` *wht*	The current location of the window's upper left corner and the size of the window in paper coordinates.
`Page at (`*minh*`,`*minv*`)` *pwd* `by` *pht*	The location of the page rectangle's upper left corner and the size of the rectangle. (The page rectangle is the smallest rectangle enclosing all rules and characters on the current page).
`IN`	The current unit of measure — one of `IN`, `CM`, `MM`, `PT`, `PC`, and `PX` (initially inches).
status	Line for status and error message display (initially blank).
`Command:`	Line for command entry.

DVI File Magnification The `/magnification=`*i* qualifier allows you to replace the magnification used in the DVI file with some other value; *i* is a positive integer 1000 × the desired magnification. The given value should be chosen carefully so that the new font sizes still correspond to existing PXL files. You should only supply a replacement magnification if you intend to print the DVI file with the same override.

Font Directory The `/font_directory=`*list* qualifier selects a master directory containing subdirectories of font PXL files. DVItoVDU gets all its font information from PXL files. *list* is a list of values separated by commas and enclosed in parentheses. A typical default list might be `(d1:[local.fonts],d1:[tex.fonts])`.

Dummy Font The `/dummy_font=`*string* qualifier specifies a font to be substituted when a font cannot be found at the size required by your document. DVItoVDU will warn you if your document uses a font at a non-existent size. Rather than abort, it will load the PXL file specified by `/dummy_font` and continue so you can look for more errors. Paragraphs using this dummy information are likely to have ragged right margins. A typical default dummy font might be `d1:[tex.fonts.1200]amr10.pxl`.

Help File The `/help_file=`*string* qualifier specifies a file containing the text to be displayed by the `?` command. Our default file is `d1:[tex]dvitovdu.hlp`.

VDU Dialogue Region

If your command line is correct and if the `/vdu` value matches the type of terminal you're actually using, then DVItoVDU will clear the screen and display something similar to Figure 1. These four lines represent the "dialogue region." The rest of the screen is called the "window region" and should be blank at this stage.

The top two lines show status information. Until a page is selected, most of the status values are meaningless and set to zero.

The third line is initially blank. DVItoVDU displays messages of various kinds in this line. Some of these messages appear only briefly but may convey helpful information. Others are more important and indicate some sort of problem, such as an invalid command or a page that won't fit on the paper; in these cases DVItoVDU will prompt you to hit the `RETURN` key before continuing.

The last line in the dialogue region is for entering commands. The first thing you normally want to do is choose a particular page for display. For example, typing '1' will select the first page in the DVI file. Many commands can be entered in the one command line. Hit RETURN to execute the command(s).

VDU Window Region

A paper pixel can be either black (corresponding to a tiny blob of ink) or white (no ink). A typical DVI page contains characters from one or more fonts, and perhaps a few rules. A rule is simply a rectangular region of black pixels, usually in the shape of a thin horizontal or vertical line. A character is usually a more complicated pattern of black and white pixels. Every character and rule has a paper position—or reference point—defined by a pair of pixel values (h,v) where h is the horizontal coordinate and v is the vertical coordinate. DVItoVDU uses a paper coordinate scheme in which the position (0,0) is a pixel one inch in from the top and left edges of the paper. Vertical coordinates increase down the paper, horizontal coordinates increase to the right. Confused? Figure 2 may help clear things up.

The window region is used to view the current DVI page. DVItoVDU treats this region of the VDU screen as a two-dimensional array of dots, but we refer to these dots as "screen pixels" to distinguish them from the paper pixels described above. A screen pixel is usually the smallest possible area on a VDU screen that can be drawn or erased; the greater the number of screen pixels in a given area, the higher the resolution of the VDU. (DVItoVDU's definition of a screen pixel is more precisely known as an "addressable location.")

The initial window sizes (widths by heights in pixels) for the VDUs currently implemented are:

AED483	512 by 442
AED512	512 by 471
ANSI	80 by 20
REGIS	768 by 400
VIS500, VIS550	1024 by 688
VT100132	132 by 20
VT640	1024 by 650

The most accurate representation of a page will occur at these unscaled values, since one paper pixel equals one screen pixel. You can increase or decrease the area currently visible by using the H and V commands to change the size of the window region. The higher the resolution of the VDU, the greater the accuracy of such scaled displays.

Note the very low resolution of the ANSI and VT100132 VDUs. These are not really graphic terminals; DVItoVDU has to define a screen pixel to be an entire character position, since the individual dots making up characters cannot be turned on and off. An ANSI screen typically consists of 24 lines of 80 columns, therefore the initial window region is 80 pixels wide and 20 pixels high (the top 4 lines are used for the dialogue region). At more useful window sizes the resulting displays will be extremely crude. Nevertheless, a variety of formatting errors can still be detected on such terminals; just don't try proofreading your document!

The size and location of the window region are automatically set every time a page is selected. DVItoVDU tries to show as much of the paper (and presumably the page) as possible, and without too much distortion. After comparing the shape of the paper with the shape of your VDU's *initial* window region, and depending on the location of the page, DVItoVDU may show the entire paper, or the top or bottom half, or the left or right half. If any part of the page is off the paper then the entire paper *and* the entire page will be shown. Since most paper sizes have portrait dimensions (width < height), and most VDU screens have landscape dimensions (width > height), DVItoVDU will normally show the top half of a sheet of paper containing the selected page.

Every time the window region needs to be updated, the entire screen is first erased.

DVItoVDU Commands

In response to the 'Command:' prompt you can enter one or more of the following commands in upper- or lowercase. Multiple commands are processed in the order given but the window region is only updated, if necessary, at the end. For example, NFD gets the Next page, switchs to Full display mode, moves the window Down, and only then displays the page. If an invalid command is detected, any further commands are ignored. Most commands consist of only one or two characters; some can be followed by parameters. Spaces before and after commands and parameters are optional.

The DVItoVDU commands are summarized in Figure 3, and they are described in detail in the sections that follow.

Miscellaneous Commands

Help The ? command displays help on the available commands.

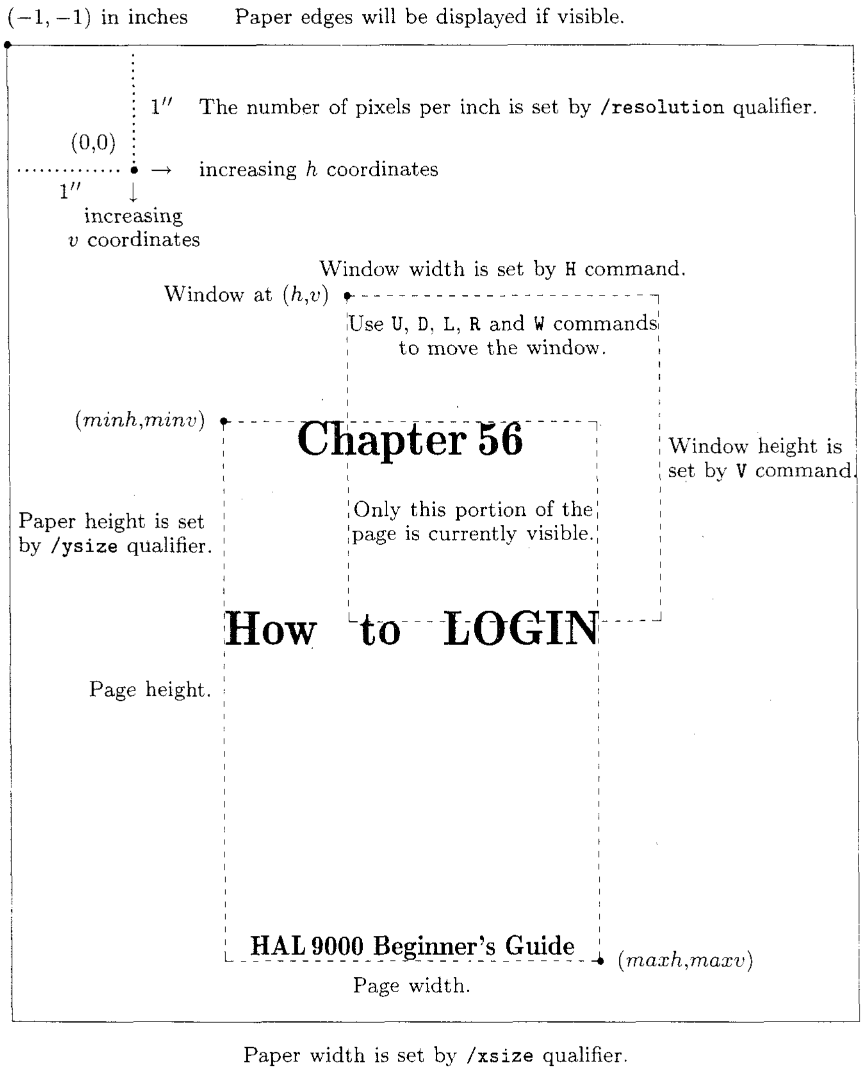

(−1, −1) in inches
Paper edges will be displayed if visible.
1″
The number of pixels per inch is set by /resolution qualifier.
(0,0)
increasing h coordinates
1″
increasing v coordinates
Window width is set by H command.
Window at (h,v)
Use U, D, L, R and W commands to move the window.
(minh,minv)
Chapter 56
Window height is set by V command.
Only this portion of the page is currently visible.
Paper height is set by /ysize qualifier.
How to LOGIN
Page height.
HAL 9000 Beginner's Guide
(maxh,maxv)
Page width.
Paper width is set by /xsize qualifier.

Figure 2.
DVItoVDU's Paper Coordinate Scheme.

Status The S command shows various qualifier values and statistics about the number of fonts, characters and rules used on the current page.

Quit The Q command quits from DVItoVDU.

Page Selection

Selection by Page Position The i command selects the ith DVI page. i must be a positive integer from 1 to n where n is the total number of pages in the DVI file.

Selection by Page Counters The [i_0.i_1. ⋯ .i_9] command selects the DVI page whose ten TEX page counters match the given specification. i_0 to i_9 are integers separated by periods. Each integer is optional and trailing periods may be omitted. An absent integer will match any value in the corresponding counter. If more than one DVI page matches, the lowest will be chosen. For example, [] is equivalent to [.........] and will select the first DVI page, even though the request matches every possible page. If your TEX source file doesn't change the value of \count0 (\pageno in plain TEX) then the ith DVI page will match TEX page [i].

Next Page The N command selects the next DVI page, depending on the current DVI page and the current direction (> or <). Before any page has been requested, N will select the first DVI page if the current direction is >, or the last DVI page if the current direction is <.

Set Forward Direction The > command arranges for future N commands to select DVI pages in ascending order. If i is the current DVI page, an N command will get page $i + 1$ unless i is the last DVI page.

Set Backward Direction The < command arranges for future N commands to select DVI pages in descending order. If i is the current DVI page, an N command will get page $i - 1$ unless i is 1.

Changing the Page Display

The way in which the current page is displayed can be varied from a full, accurate representation to a terse, fast display for when fine details are unimportant. The window region is updated in the following manner: Visible paper edges are drawn first followed by visible rules (shown in full no matter what the display mode). Visible characters are finally shown on a font by font basis; those fonts with the least number of characters on the page are drawn first. Every few rules or characters, DVItoVDU will check to see if you've typed something at the keyboard; you can hit the RETURN key to abort the display, or you can change the display mode by hitting the T, B or F keys.

Terse Character Display The T command displays a terse representation of characters. On most VDUs the TEX text fonts should be readable; the characters will be in approximately the right position and may even be about the right size. Note that the text fonts will all look alike; you won't be able to distinguish between roman and bold characters for example. Most VDUs assume all characters come from a TEX text font and map them into similar-looking ASCII characters. Characters from non-text fonts, such as math symbols, will usually appear incorrect.

Outline Box Character Display The B command displays box outlines of the smallest rectangles containing all black pixels in characters. The reference points of most TEX characters are usually located near the bottom left corners of these boxes. Box mode is intermediate in speed between Terse and Full modes.

Full Character Display The F command displays a full representation of all pixels in characters. This display is the most accurate but may take some time; hit RETURN or switch to Terse or Box mode if you get bored. On the higher resolution VDUs, a good compromise between speed and accuracy is to start off in Full mode so that math symbols and any other special characters are displayed correctly, and to switch to Terse mode when the bulk of the text begins.

Changing Units of Dimensions

All the numbers in the second line of the dialogue region are dimensions in terms of the units shown at the end of the line. The parameters following some commands are also dimensions in terms of these units. Unlike the dimensions in TEX, you don't explicitly type the units when you need to specify a dimension to DVItoVDU; simply enter an integer value or real value (which will be truncated to four decimal places if necessary). A given value is rounded up internally to the nearest paper pixel based on the current units and the conversion factors shown below.

Inches The IN command causes dimensions to be shown and entered in terms of inches (/resolution defines the number of paper pixels per inch).

Centimetres The CM command causes dimensions to be shown and entered in terms of centimetres (2.54 cm = 1 in).

Millimetres The MM command causes dimensions to be shown and entered in terms of millimetres (10 mm = 1 cm).

Picas The PC command causes dimensions to be shown and entered in terms of picas (1 pc = 12 pt).

Points The PT command causes dimensions to be shown and entered in terms of points (72.27 pt = 1 in).

Pixels The PX command causes dimensions to be shown and entered in terms of paper pixels.

Moving the Window

The window region can be moved to any position over the current page. The parameters h and v are dimensions ranging from −480 inches to +480 inches (for those of you TEXing billboards). You will be told if the entire window moves outside the page rectangle defined by *minh*, *minv*, *maxh* and *maxv*. If this does happen, the movement is restricted to *just outside* the edges to make it easier to get back over the page using only the U, D, L and R commands. Note that the location of the window is automatically set every time a page is selected. This position will normally be $(-1,-1)$ in inches; i.e., the top left corner of the paper.

Set Window Position The `W` h,v command moves the window region's top left corner to the given paper position. h is the horizontal coordinate, v is the vertical coordinate. If h and v are absent then the window is moved to (*minh*,*minv*), the top left corner of the page rectangle.

Up The `U` v command moves the window up v units. If v is absent then window moves up by its current height.

Down The `D` v command moves the window down v units. If v is absent then window moves down by its current height.

Left The `L` h command moves the window left h units. If h is absent then window moves left by its current width.

Right The `R` h command moves the window right h units. If h is absent then window moves right by its current width.

A positive integer following a command that can take a parameter is never interpreted as a DVI page selection. For example, '`D5`' will always be interpreted as "move the window down 5 units" and never as "move the window down by its current height and then select page 5." As it turns out, this ambiguity is not a problem because all the commands that can have parameters only affect the *current* page. There is no point in moving the window (or changing its size) and then selecting a page in the same command line since the window location is automatically reset every time a page is selected.

Changing the Window Size

The width and height of the window region can be changed independently. It is up to you to maintain a suitable aspect ratio. The location of the window will not change unless the page becomes invisible. The parameters *wd* and *ht* are dimensions ranging from 1 pixel to 480 inches. Note that the width and height of the window are automatically set every time a page is selected; their values will normally depend on the paper dimensions.

Horizontal Size The `H` *wd* command sets the Horizontal size of the window to the given width. If *wd* is absent then window width is set to its initial, unscaled value.

Miscellaneous

`?`	Show help on commands.
`S`	Show qualifier values and current page status.
`Q`	Quit.

Page Selection

i	Select ith DVI page.
`[`i_0`.`i_1`.` $\cdots$ `.`i_9`]`	Select DVI page according to TEX page counters.
`N`	Select next DVI page.
`>`	Make `N` command move forward.
`<`	Make `N` command move backward.

Page Display

`T`	Select terse character display.
`B`	Select bounding rectangle character display.
`F`	Select full pixel character display.

Units of Dimensions

`IN`	Use inches.
`CM`	Use centimetres.
`MM`	Use millimetres.
`PC`	Use picas.
`PT`	Use points.
`PX`	Use pixels.

Moving the Window

`W` h,v	Move top left corner to (h, v).
`U` v	Move up v units.
`D` v	Move down v units.
`L` h	Move left h units.
`R` h	Move right h units.

Changing Window Size

`H` *wd*	Set horizontal size to *wd*.
`V` *ht*	Set vertical size *ht*.

Figure 3. *Summary of Commands.*

Vertical Size The `V` *ht* command sets the Vertical size of the window to the given height. If *ht* is absent then window height is set to its initial, unscaled value.

Just a brief note on the scaling method used by DVItoVDU: Rules and glyphs having the same top, bottom, left or right paper coordinates also have the same scaled coordinates. This ensures baselines will line up but means that rules and glyphs may change shape when the window is moved to a new position. The effect is most noticeable on the low-resolution VDUs.

Command Examples

The spaces between commands in the following examples are for clarity; they are not mandatory. Commands can also be typed in lowercase.

`< N N N`

If these commands are given before any page has been requested, they will select the 3rd last page (assuming there are at least 3 pages). Note that DVItoVDU will only display the 3rd last page. The intervening pages are still processed though, and you'll be warned about any problems with them (such as a page off the paper).

`F W H V`

This command sequence can be very useful for looking carefully at the results of a small TEX experiment. You might, for instance, want to check the appearance of two characters moved closer together by a negative `\kern`. Remember that the unscaled window size chosen by `HV` produces the most accurate display, since each screen pixel corresponds to exactly one paper pixel.

`R9999 L D9999 U`

Sometimes you need to move quickly to the right edge of the page to have a look at line breaks, or you might want to go to the bottom and look at where the page was broken. This particular command sequence will move the window's *bottom right* corner to the bottom right corner of the current page (*maxh*,*maxv*). `R9999` moves the entire window to the right of the page, but only just. `L` then moves the window left by its current width so that the *right* edges of the window and the page coincide. `D9999` moves the entire window below the page, but only just. `U` then moves the window up by its current height so that the *bottom* edges of the window and page also coincide.

`IN W-1,-1 V12 H16 T`

It is often useful to get an overview of the positioning of the page within the entire paper. These commands will do just that on all the VDUs currently implemented (assuming A4 paper dimensions). `IN` sets the current units to inches just in case they were something else and `W-1,-1` moves the window to the top left corner of the paper. `V12` sets the window height to slightly more than A4 paper height and `H16` sets the window width to a value that will ensure the paper shape maintains approximately the right proportions. `T` sets the display mode to Terse since you're probably not concerned with finer details when looking at the entire paper.

`[.56] W IN H2.5 V2.6 R1 U1 F`

This is the likely command sequence that led to the window display shown in Figure 2. The author has been clever enough to define a `\chapter` macro that sets `\count1` to the given chapter number. `[.56]` will thus select the appropriate DVI page (`\count0` is ignored).

Description of System Design

Most of the material in this section is from the *DVItoVDU System Guide.*

As its name would imply, a Modula-2 program is typically built up from a number of separately compiled modules. Each module can import data and procedures from other modules. An imported module is further decomposed into a definition module and an implementation module. The definition part contains declarations for all exported objects and serves as the interface to client modules. The details of how exported objects are actually realized are hidden from clients in the implementation part. Rapid system regeneration is possible because client modules depend only on the definition part; the implementation part may be modified (e.g., optimized) without the need to recompile any client modules. Figure 4 shows the importation dependencies of all the modules making up DVItoVDU.

Not only do modules allow a large program to be broken up into more manageable chunks, they also provide a sensible basis for a description of that program:

DVItoVDU The main module is primarily concerned with the user interface. It handles all the interactive command processing and contains the high-level logic used to update the dialogue and window regions. Data and procedures are imported from the following modules to help the main module carry out its many tasks.

DCLInterface This module provides the interface to the VAX/VMS command language interpreter. (DCL stands for Digital Command Language, just in case you're wondering.) The command line used to invoke DVItoVDU is parsed and the DVI file name extracted. All the qualifiers are also initialized, either to explicit values given in the command line or to site-dependent default values.

VDUInterface DVItoVDU can work efficiently on different types of VDUs by letting Modula-2 procedure variables act as generic VDU routines. These routines, along with the generic VDU parameters, are defined in *VDUInterface.* As little as possible is assumed about the capabilities of a VDU. In particular, the availability of a graphic input device is ignored. DVItoVDU should be able to work on any terminal that can:

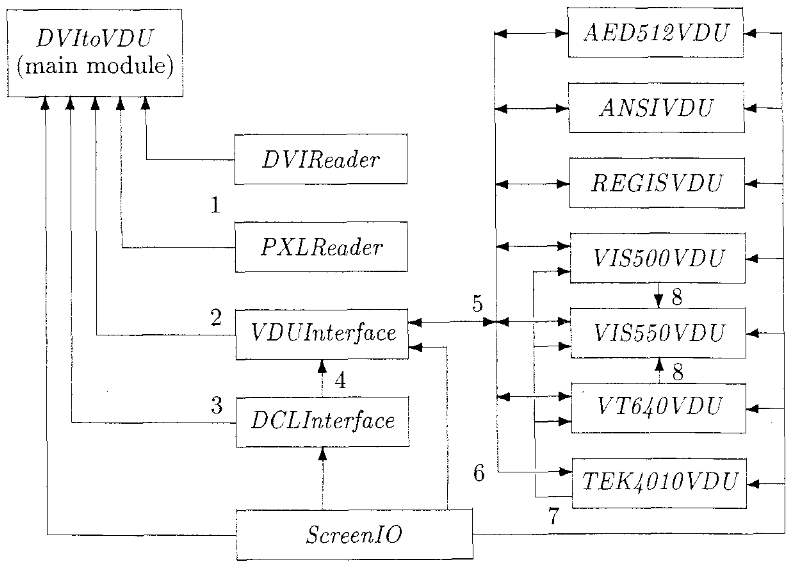


Figure 4.

DVItoVDU's Module Map. An arrow from module A to module B indicates the latter imports data and/or procedures from the former (B is said to be a client of A). If the definition part of module A changes, then all client modules must be recompiled and DVItoVDU relinked. If the implementation part of module A changes, only the relinking stage is necessary.

Points of interest:

1. *DVIReader* does not depend on *PXLReader*. It does know that PXL files contain crucial typesetting data, but leaves it up to the main module to specify how and where to get such information.
2. *VDUInterface* exports the generic VDU routines and parameters used in the main module.
3. *DCLInterface* extracts the DVI file name and qualifiers from the VMS command line.
4. *VDUInterface* needs the `/vdu` value to select an appropriate VDU initialization routine.
5. Specific VDU modules import the generic VDU routines and parameters from *VDUInterface* and, in return, export an initialization routine. *VDUInterface* will decide which one is actually executed.
6. The Tektronix 4010 module imports the *TeXtoASCII* array to map TEX characters into corresponding ASCII characters. Note that *VDUInterface* does not know about the Tektronix 4010 module.
7. The VISUAL 500, VISUAL 550 and VT640 terminals all emulate Tektronix 4010 graphics.
8. The VISUAL 550 terminal uses VISUAL 500 graphic routines to update the window region and VT640 routines to update the dialogue region.

- Mix text and graphics on the screen (some VDUs make no distinction).
- Erase all of the screen, or individual text lines.
- Move the cursor to any given screen pixel.
- Display a rectangular region of screen pixels (possibly just one).

The generic VDU routines are:

StartText	switch to "text mode"
ClearTextLine	erase given line
MoveToTextLine	move to start of given line
ClearScreen	erase the entire screen
StartGraphics	switch to "graphics mode"
LoadFont	for later *ShowChar* calls
ShowChar	show given Terse character
ShowRectangle	show given rectangle
ResetVDU	may need to reset VDU

Most are quite trivial to implement for a specific VDU. The main module looks after all the tricky graphic operations such as the clipping of characters and rules outside the current window region, and the way in which visible paper pixels are scaled to screen pixels.

From an efficiency point of view, the two most critical routines are *ShowChar* and *ShowRectangle*. *ShowChar* is used by the main module to display a character in Terse mode. The only information given is the TEX character (currently restricted to `\char0..\char127`) and its screen position. Since characters are displayed one font at a time, some VDUs can use scaling information sent by the most recent *LoadFont* routine to select an appropriate hardware font.

ShowRectangle is used for all other window graphics. It is used to draw the paper edges, to draw all rules (regardless of display mode), to draw all glyph outlines in a Box display, and to draw the horizontal lines making up all glyphs in a Full display. The majority of rectangles are in fact horizontal or vertical lines just one screen pixel thick. The generic VDU parameters are:

DVIstatusl	DVI status line, usually 1
windowstatusl	window status line, usually 2
messagel	message line, usually 3
commandl	command line, usually 4
bottoml	bottom text line in screen
windowh	window's top left h coord
windowv	window's top left v coord
windowwd	unscaled window width
windowht	unscaled window height

These parameters are actually integer variables. The main module treats them as constants.

There are two screen coordinate systems used by DVItoVDU:

■ When updating the screen in text mode (i.e., when updating the dialogue region or during a `?` or `S` command), DVItoVDU assumes text lines start at 1 and increase downwards. The bottom text line on the screen is given by the parameter *bottoml*.

■ When updating the screen in graphics mode, DVItoVDU assumes the top left screen pixel is at (0,0). Horizontal coordinates increase to the right and vertical coordinates increase down the screen. The top left pixel in the window region is at (*windowh*,*windowv*). Specific VDU modules may have to do a translation to the actual coordinate scheme used by the VDU. The size of the window region in screen pixels is given by *windowwd* and *windowht*.

AED512VDU, ANSIVDU, REGISVDU, ... Each specific VDU module exports an initialization routine that will assign appropriate procedures to the generic VDU routines and specific integers to the generic VDU parameters. *VDUInterface* imports all these initialization routines and uses the `/vdu` value set in *DCLInterface* to execute one of them.

DVIReader This module exports the routines and data structures needed to move about randomly in a DVI file and interpret selected pages. Although the main module is currently the only client, it is anticipated that *DVIReader* could just as well form the basis of a more conventional DVI translator such as a non-interactive device driver.

Font, character and rule information is stored in dynamically allocated lists to avoid imposing any limit on their numbers. The length of the font list is determined soon after opening the DVI file by reading all the font definitions in the postamble. Each font node is a record made up of many fields; one of these fields is the head of a character list. The nodes in each character list will store the positions and TEX codes of all characters on a page. Besides the font list, there is also a rule list. The nodes in the rule list will store the positions and dimensions of all rules on a page. (Character and rule positions are stored as pairs of horizontal and vertical paper pixel coordinates. The manner in which *DVIReader* calculates such positions is based firmly on Donald Knuth's `DVItype`.)

Just before interpreting a selected DVI page, the rule list and all the character lists are deallocated if necessary. During interpretation, *DVIReader* adds a new rule or character node to the *tail* of an appropriate list. When the main module processes such lists, rules and characters will be displayed in somewhat the same sequence as seen in the DVI page; i.e., top-to-bottom and left-to-right. (Since there is a separate rule list, as well as a character list for each font, the precise sequence is not remembered.) After interpretation, the nodes in the font list are sorted so that fonts with the least number of characters (> 0) will be processed first.

The various lists contain most of the information needed to display the page; they are traversed by the main module whenever the window region is updated. If the page isn't empty then *DVIReader* will also determine the edges of the page rectangle. This is the smallest rectangle containing all black pixels in glyphs and rules, as well as all character reference points (needed for Terse displays). The main module uses the page rectangle to decide if the page is off the paper, and to restrict window movement.

PXLReader DVItoVDU gets all its character information from standard PXL files. *PXLReader* exports routines for moving about in such files and grabbing various bytes and words. A PXL file contains the crucial TFM widths needed by *DVIReader* to correctly position characters when interpreting a DVI page. These widths, along with other details about each glyph, are kept in a PXL file's font directory. A directory is loaded just once for each font (the very first time the font is seen) and DVItoVDU will display '`Loading font data from ...`' in the message line.

A PXL file also contains glyph shape information (in the form of bitmaps) for all characters in a TEX font. The main module uses these bitmaps during a Full display to draw characters a font at a time. Each time a PXL file is opened, the message '`Drawing characters from ...`' will appear.

Because *DVIReader* creates a separate character list for each font used on a page, there never needs to be more than one PXL file open at any given time.

ScreenIO All low-level terminal i/o is handled by the routines defined in this module.

Performance

A number of methods are used to make DVItoVDU an efficient program:

- Output buffering reduces the number of calls to the standard VMS terminal output routine.
- The DVI file and associated PXL files are mapped into virtual memory for fast random access.
- When a glyph is vertically scaled down during a Full display, overlapping rows in its bitmap are first ORed together to reduce the number of *ShowRectangle* calls needed to build up the glyph.
- Since the reference points of most characters in a line will have the same vertical coordinate, some *ShowChar* implementations reduce the number of output bytes needed to update the screen position from one character to the next by remembering the last vertical coordinate. (The sequence of *ShowChar* calls for each font is determined by the way *DVIReader* builds a character list.)
- Specific *ShowChar* routines map a given TEX character into a similar-looking ASCII character using *TeXtoASCII*, a look-up table imported from *VDUInterface*. (Since all TEX characters are assumed to come from text fonts, those from non-text fonts will appear incorrect.)

The following table shows the times taken to display an entire DVI page on different VDUs:

	Terse Display		Full Display	
	compute	elapsed	compute	elapsed
`VT220`	0.9	10	7.3	56
`VT640`	1.2	15	13.9	145
`AED512`	?	23	?	159
`GIGI`	1.5	34	14.6	2488
`VIS500`	1.6	52	18.2	193

all times are in seconds

The figures for the VISUAL 500 were obtained while running the program on a VAX-11/780 with 20 interactive users. All the other VDUs were on a VAX-11/785 with 15 to 20 users. Each terminal was operating at 9600 baud.

The page used in the above benchmark contained some 2,500 characters from 16 fonts. The window size was chosen by typing '`H16V12`' in each session. Substantial improvement in performance is unlikely for any of the current batch of VDUs. The large discrepancies between computation and elapsed times, particularly in a Full display, are mostly due to the huge number of bytes that must be sent to a VDU to update the window region.

The times shown in the above table do not include page interpretation. Most pages can be translated in about a second of elapsed time, depending on how busy your VAX is and how many fonts are seen for the very first time. For example, the first 50 pages in the preliminary LATEX manual (the December 1983 version) were interpreted by typing '`NNN...`'. This took just over a minute of elapsed time on a VAX-11/785 with 17 users logged in (compute time was 22.3 seconds). Some 35 fonts were loaded. When the same 50 pages were processed again (by typing '`1NN...`'), the elapsed time dropped to 30 seconds and the compute time to 15.9 seconds.

Conclusion

The current version of DVItoVDU should remain quite stable, particularly from the user's viewpoint. (I am open to complaints or suggestions though!) Any changes are more likely to occur behind the screens, so to speak. One change already in sight is the reading of font data from GF files instead of PXL files.

Other VAX/VMS sites with Modula-2 may be sufficiently motivated to get the program running on new types of VDUs. An implementation on a state-of-the-art, bit-mapped graphics terminal would be something to behold. Note that a new VDU can be added without having to make any changes to the main module. The *DVItoVDU System Guide* lists the necessary steps.

It should be pointed out that DVItoVDU is currently targeted towards relatively primitive terminals. Significant changes to both the display logic and page data structures would probably be needed to take full advantage of the latest graphic workstations. For example, a large amount of bit-mapped memory could store an entire rasterized page and allow much more sophisticated updating of the window region. Pan and zoom operations would also be much easier using a mouse or thumbwheel cursor controls.

A few hardy souls may even like to translate DVItoVDU into another language or transport it to another operating system. Note, however, that Modula-2 does not specify any i/o facilities as part of its language definition. While such facilities are usually provided as library functions, there is currently no widely accepted standard library. Until this happens, Modula-2 programs will not be very portable.

To help make any conversion easier, all system-dependent code can be quickly located by searching for the string "`SYSDEP`" in the various source files. Terminal i/o is isolated in the *ScreenIO* module and most file operations are confined to the *DVIReader* and *PXLReader* modules. *DCLInterface* is also highly VAX/VMS-dependent. I can only encourage such attempts by stressing the advantages of a TEX page previewer:

- Paper usage is reduced.
- Document preparation time is reduced. The typical proofing cycle of

$$\begin{tikzcd} & \text{\TeX} \arrow[dr] & \\ \text{edit} \arrow[ur] & & \text{print} \arrow[ll] \end{tikzcd}$$

can be replaced by

$$\begin{tikzcd} & \text{\TeX} \arrow[dr] & \\ \text{edit} \arrow[ur] & & \text{preview.} \arrow[ll] \end{tikzcd}$$

- Experimentation is encouraged. The results of changing a global formatting parameter or altering a macro definition can be quickly seen and evaluated.

MacDraw Pictures in TeX Documents

Hal Varian
University of Michigan
Jim Sterken
Textset, Incorporated

MacDraw is a program available for the Apple Macintosh that allows one to generate "technical" drawings. MacDraw uses an object oriented approach to graphics design which is suitable for many illustration needs. MacDraw should not be confused with MacPaint, which is a bitmap oriented system for "artistic" drawing. MacPaint represents a drawing by bit images, and thus its resolution is limited to the resolution of the Mac screen. MacDraw represents a drawing by a list of the objects involved (square, oval, line, etc.) and their location on the page. Its resolution is determined by the resolution of the output device.

MacDraw is capable of generating PostScript files which can be used to generate high resolution (300 dpi) images on the Apple LaserWriter and other PostScript printers. Textset has developed a `DVI` to PostScript translator, `DVILASER/PS`, that allows TeX `DVI` files to be printed on a PostScript printer. It seemed to us that it would be useful to be able to merge the graphics images generated by MacDraw with the text output generated by TeX. In this note we describe how we were able to accomplish this.

An Overview

Once the system is in place it is quite easy to use. First you run MacDraw on the Mac and create your drawings. Then you create the PostScript files on the Mac, and transfer them to the computer on which you will be running `DVILASER/PS`. (We did most of our work on an IBM AT, but `DVILASER/PS` also runs on a number of other operating systems and the same procedure should work for them.) You then use TeX's `\special` command to insert the MacDraw created files into your document. When the document is printed you will have nicely integrated text and graphics.

Despite the ease with which the system can be used, there are several tricks that we had to figure out to get everything working together smoothly. We're describing the tricks here, so that others can benefit from our experience. We will assume that the reader is familiar with MacDraw and the general operating principles of the IBM AT and Macintosh.

Generating PostScript Files

Prepare your MacDraw disk by installing the LaserWriter icon in the system. You do not need to have the LaserPrep file on this disk, but you will need that file later.

Start MacDraw and create an illustration in the top 3 inches of the page. This is basically the area shown on the Macintosh screen. The TeX macro defined below expects to find the MacDraw drawing in this location. Of course it can be rewritten to find it elsewhere or to scale or rotate the MacDraw drawing before merging it into the TeX document, but we leave that up to you.

When you have completed the drawing, it is a good idea to save it in case something goes wrong. Once you have done that you can create the PostScript file which describes the drawing. To accomplish this you must begin by making sure you have selected the LaserWriter printer using the "Select printer" desk accessory. Then open MacDraw and choose the "Print" option. When the LaserWriter menu box comes up, you hold down the cloverleaf key and the "F" key simultaneously. At this point the Macintosh should start to sound a bell and part of the display will flash. Keep holding down the two keys and click the "OK" box. Don't let up

Editor's Note: this article was originally printed on an Apple LaserWriter using the LaserWriter's Times Roman fonts, with the illustration in Figure 2 inserted by the DVILASER/PS program. For this special issue of *TUGBOAT*, it was reformatted, typeset on an Alphatype CRS using the new Computer Modern fonts, and the illustration was reinserted manually.

pressure on the two keys until a box appears that says "Creating PostScript File."

The Mac will then create a file that contains the PostScript commands that it would normally send directly to your LaserWriter. It stores these commands in a text file called "PostScript." There seems to be no way to get it to use a more informative name. Of course, you can use the Finder to rename the file to something more meaningful. Since you are going to transfer this file to the PC, it is a good idea to give it a legal MS–DOS name. However, to rename the file you have to exit MacDraw. This is not a great problem if you are only converting a few files, but if you are converting many files, it can take a lot of time. We used a public domain desk accessory called "Rename" which allowed us to rename the PostScript file from within MacDraw. There are other desk accessories available that will also rename files, but we stuck with the first one we found that worked.

The properly named file can then be incorporated, as is, into your TEX document via the `\PrintMacDraw` macro defined below. The `\PrintMacDraw` macro assumes that figures are being prepared in MacDraw to print at the very top of the page — it then positions them to the correct spot from there.

You should be warned that this way of generating PostScript files is an undocumented debugging feature of the Mac. It seems to work with other Mac programs as well, but since it is undocumented, we have no idea if it will continue to work in future releases of Mac software.

Sending the File to the IBM AT

Now you need to transmit your PostScript file to the computer that you will be using to run `DVILASER/PS`. We used an IBM AT, but similar procedures should work for other computers.

We used `KERMIT` to transfer the PostScript file from the Mac to the AT. There are other systems available that should work equally well, but we'll describe our `KERMIT` experience.

First, you need a cable. It turns out that the Imagewriter cable that connects the Mac to its printer will work fine for connecting a Mac to an IBM PC, if the serial port on the PC has the right gender. If not, you need a gender converter as well. If you are using an IBM AT, it has a 9 pin serial port just like the Mac but — surprise! — they are not configured in the same way. We ended up using a standard AT-to-modem cable, plugged into a gender converter, plugged into the Imagewriter cable.

Now start up `KERMIT` on both systems and make sure that each is configured at the same baud rate, parity, etc. We used 9600 bpi and no parity. Now set the Mac to "Be a server" and move to the AT and type "get *filename*" where *filename* is the name of the PostScript file on the Mac that you want to transfer. If all goes well, the file will be transferred over. Again, there are other procedures that you can use, but we found putting the Mac in the server mode and using "get" was the quickest and easiest for us. If you have your PostScript files on a Mac drive other than the default drive, you can use the "send file" menu to change the default drive before using "get."

Preparing the LaserWriter

Now you are almost ready to print your document. But first you have to prepare the LaserWriter to accept the MacDraw document. Remember the file LaserPrep that comes with your LaserWriter? This file consists of a number of PostScript procedures that are used by MacDraw. You have to send this file to the LaserWriter before it can understand the material in your MacDraw file.

Unfortunately, the LaserPrep file is not a straight ASCII file. It seems to consist of a Mac program that contains embedded in it the ASCII codes of the PostScript macros. In order to use the AT to send these commands to the LaserWriter, you have to convert the LaserPrep file to straight ASCII.

By far the easiest way to do this is to find someone else who has already done it. We found a copy of the LaserWriter header file on Info–Mac. According to the note attached to it, "This is not the official Apple header file. It is neither endorsed nor condemned by Apple." If you can get hold of this file, it will make things a lot easier. A copy of this file is included with `DVILASER/PS`.

If not, you can still create one for yourself, but it will take more work. Here is how you do it. First, get hold of the Shareware editor `FEDIT`, created by John Mitchell. This is a handy little program for editing Macintosh files. Use `FEDIT` to examine a copy of the LaserPrep file.

Now pull down the "Options" menu and choose "Reverse Forks." This will make the Macintosh think that LaserPrep file is a data file rather than a resource file. When you've done this you can exit from `FEDIT`.

Next use `KERMIT` to transfer LaserPrep to the IBM. From now on, we'll refer to this file as `laser.prp`. Use your favorite text editor to clean up LaserPrep. Here is what you have to do. First, get rid of the binary junk at the beginning and end of the file. Next, get rid of the random characters at the end of the PostScript lines in this file. Finally, comment out the first few lines of the file up to but not including the line that begins `/md 200`. These lines contain commands causing the LaserPrep procedures to be loaded "permanently" in the LaserWriter — until the power is turned off. With the approach we used it was not necessary to load the LaserPrep procedures permanently.

Now we'll assume that you have a file called `laser.prp` that contains the PostScript definition used by MacDraw. There are still two small modifications you have to make to it. First, you should comment out the lines that say:

```
userdict /note known
{{legal}{note}ifelse}
{pop}
ifelse
```

These concern the kind of paper used and we're assuming that you use only 8.5 by 11 paper. If they are not removed they blank the current page causing problems when merging several MacDraw files onto the same page.

Next, add a single line with the command `pop` at the end of the `laser.prp` file. This will clean up the PostScript stack.

That's it. You now have the `laser.prp` ready to send to your LaserWriter. As newer versions of LaserPrep are released by Apple other changes will inevitably be necessary, so be prepared to learn a little PostScript. This will continue to be a problem until Apple decides to officially support the generation of PostScript files from MacDraw in some fashion.

Inserting the File into Your Document

The TEX macro `\PrintMacDraw`, whose definition is given in Figure 1, uses TEX `\special` commands to embed in the `DVI` file the necessary commands to cause `DVILASER/PS` to insert the PostScript commands for your MacDraw drawing into the PostScript file it creates.

The `\PrintMacDraw` macro takes the usual approach used to merge graphics into TEX documents. It first makes an empty `\vbox` to reserve space for the figure. Then it uses a variety of `DVILASER/PS` `\special` command options to generate "inline" PostScript commands which configure the PostScript co-ordinate system to match that required by MacDraw. Finally, another `plotfile` command is used to include the unchanged MacDraw PostScript file in the space that was

```
%
% PrintMacDraw parameters:
%  #1: name of MacDraw-generated PostScript file
%  #2: height of space to reserve
%
\def\PrintMacDraw#1#2{\par
  \vbox to #2{
    \special{ps::[asis, begin]
      0 SPB
      /MacDrawCheckPoint save def
      md begin /page {pop} def end
      Xpos Ypos translate
      0 10.75 72 mul neg translate
      }
    \special{ps: plotfile #1 asis}
    \special{ps::[asis,end]
      MacDrawCheckPoint restore
      0 SPE
      }
    \vss
    }
  }
```

Figure 1. *TEX definition of* `PrintMacDraw`. *Used in conjunction with* `DVILASER/PS`, *it inserts a PostScript drawing in a TEX document.*

reserved. The `DVILASER/PS` `\special` command options are described fully in the `DVILASER/PS` documentation.

Figure 2 was printed using an unmodified MacDraw-generated PostScript file. To accomplish this, the following `\special` command was inserted at the beginning of the document:

```
\special{ps: plotfile laser.prp global}
```

where `plotfile` is a `DVILASER/PS` command causing the contents of file `laser.prp` to be inserted into the PostScript file it is preparing. This defines the PostScript procedures referenced in the MacDraw-generated PostScript file `macdraw.tst` which was included later via the TEX macro call:

```
\PrintMacDraw{macdraw.tst}{3.75in}
```

A Few More Tips

The MacDraw material can use a fair amount of memory in the LaserWriter. `DVILASER/PS` is also using memory in the LaserWriter. Sometimes these conflicting needs cause the LaserWriter to issue a "virtual memory error" when interpreting the PostScript file. In those cases you can change the `DVILASER/PS` configuration file, `DVILASER.OPT`, to tell it to assume that less LaserWriter memory is available.

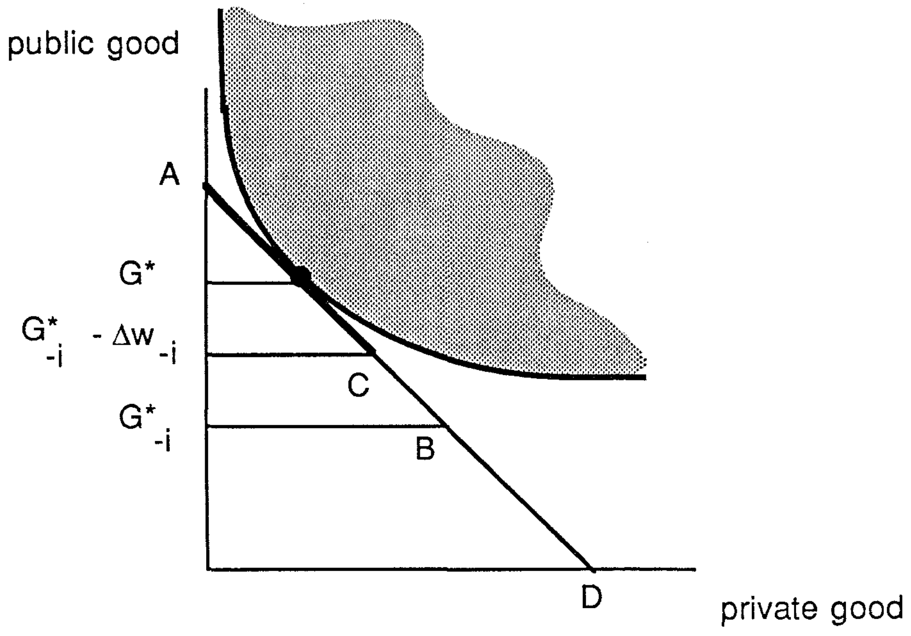


Figure 2.
A sample MacDraw picture printed by DVILASER/PS.

It is best to use `\PrintMacDraw` within a `\midinsert` so that TEX can figure out a good page placement for it. Otherwise, you can get peculiar looking output.

Conclusion

This approach to integrating TEX with MacDraw graphics has proved quite valuable as well as relatively easy to use. We expect that as more and more graphics packages include the ability to emit PostScript code directly, greater attention will be focused on the special problems of integrating TEX with graphics.

We have used these techniques to print a 550 page book with over 200 illustrations. Other than occasionally having to reconfigure the LaserWriter to reserve less memory for DVILASER/PS and more for MacDraw, no problems were encountered.

DVILASER/PS EXTENSIONS TO LaTeX

Doug Maus
Bruce Baker
Textset, Incorporated

Overview

The LaTeX Document Preparation System is a collection of TeX macros designed to make it easy to produce high-quality documents. LaTeX was developed by Leslie Lamport — also the author of *LaTeX: A Document Preparation System* which is published by Addison-Wesley Publishing Company. LaTeX is in the public domain and is included in the standard TeX distribution available from Stanford University. It is also included in most proprietary implementations of TeX.

LaTeX is based on the concept of a document as a set of structures (e.g. descriptions, tables, enumerated lists) called *environments*. Environments start with a `\begin{`*environment*`}` statement such as `\begin{tabular}` or `\begin{verbatim}` and end with an analogous `\end{`*environment*`}` statement.

This article describes an upward compatible set of extensions to LaTeX's `picture` environment implemented by Textset to take advantage of the graphics capabilities of PostScript language used by the Apple LaserWriter and other printers. Textset's TeX output driver for the PostScript printers, DVILASER/PS, includes support which makes it easy to integrate user-supplied PostScript statements with the PostScript statements being generated automatically by DVILASER/PS as it converts TeX DVI files into PostScript format. The LaTeX extensions are included free of charge with all DVILASER/PS distributions.

Editor's Note: this article was originally printed on an Apple LaserWriter using the Almost Computer Modern fonts, with the illustrations in Figure 1, Figure 3, Figure 4 and Figure 6 inserted by the DVILASER/PS program. For this special issue of *TUGBOAT*, it was reformatted, typeset on an Alphatype CRS using the new Computer Modern fonts, and the illustrations were reinserted manually.

The TeX macro package described here contains about 300 lines of code and has been dedicated to the public domain by Textset. Although the macros assume one is using Textset's DVILASER/PS program, they should be of general interest. The same strategy could be applied with other device drivers. In fact, similar extensions to LaTeX could be implemented for any printer with a reasonable graphics language.

Some of this work was funded by the University of Michigan and developed initially at the CAEN Apollo Lab, and many thanks are due to Leslie A. Olsen and our other friends at the College of Engineering.

The LaTeX Picture Environment

One of LaTeX's nicest features is the `picture` environment. With it, one can "draw" pictures. The picture in Figure 1 was created by the LaTeX commands that are to the left of it.

The "(1,1)$(-150, 27)$" specifies how much space the picture should take up and also allows the user to move the picture around on the page. In the example, "(1,1)" tells LaTeX to allocate hardly any space at all for the picture — just an imaginary box 1pt by 1pt, though obviously the actual picture is much larger. The "$(-150, 27)$" positions the picture in the desired place by moving the whole picture to the right 150pt and down 27pt from where LaTeX would have otherwise have put it, at the next place a regular character would go that has dimensions 1pt by 1pt.

Within the LaTeX `picture` environment, pictures normally are "drawn" by positioning special characters, in this case curved and straight line segments, in ways that give the appearance of circles and continuous lines. Circles are actually composed of four discrete curved segments, one for each quarter of the circle. LaTeX carefully aligns them so that

```
\begin{picture}(1,1)(-150,27)
 \put(150,200){\circle{40}} % head
 \put(140,205){\circle*{10}} % left eye
 \put(160,205){\circle*{10}} % right eye
 \put(150,193){\circle{15}} % mouth
 \put(150,180){\line(0,-1){100}} % body
 \put(150,150){\vector(-1,-1){50}} % left arm
 \put(150,150){\vector(1,-1){50}} % right arm
 \put(150,80){\line(-1,-1){50}} % left leg
 \put(150,80){\line(1,-1){50}} % right leg
\end{picture}
```

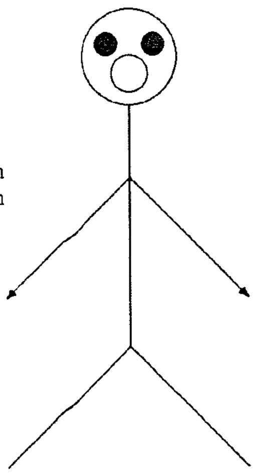


Figure 1.

Sample LaTeX picture commands, and the resulting picture.

they print as a complete circle. Oblique lines, ones that are neither vertical nor horizontal, are composed of a number of shorter segments which are placed end on end and form what looks like a continuous line. These special characters are obtained from a set of special fonts (`circle10`, `line10`, `lasy10`, etc.) that are distributed with LaTeX. Using these special fonts, LaTeX "fools" TeX into drawing pictures even though TeX doesn't really have any graphics capabilities.

LaTeX's `picture` environment is somewhat restrictive since "pictures" must be put together from the limited number of "pieces" available in the special LaTeX fonts. LaTeX carefully figures out the position of each circle or line segment. Longer lines, since they are made up of many shorter segments, cause longer execution and printing times. Even worse, creators of complex pictures often get the error "TeX capacity exceeded — sorry." This happens because TeX runs out of memory since there are too many "characters" on the page to handle. In addition, the user must be sure to use lines having only certain slopes, and be willing to accept circles of only certain diameters since in the standard LaTeX distribution there are only 36 available slopes for straight lines and 10 different diameters for circles. Also, LaTeX can only print hollow circles of diameters that are multiples of 4pt in size. If a user selects an in-between size like 21pt, LaTeX will pick the closest available size: 20pt. The biggest hollow circle it can do is 40pt — a little over half an inch — in diameter. The biggest solid circle is 15pt in diameter. The steepest non-vertical slope available is $\pm\frac{5}{6}$. LaTeX checks the slope arguments to make sure neither Δx nor Δy exceeds ±6, and that they are both integers. For vectors, because the arrowhead is drawn from a font too and slope availability in its case is even more limited, neither Δx nor Δy can exceed ±4.

```
\begin{picture}(300,300)
 \put(150,200){\circle{70}} % head
 \put(140,205){\circle*{1}} % left eye
 \put(160,205){\circle*{1}} % right eye
 \put(150,193){\circle{10}} % mouth
 \put(150,165){\line(0,-1){85}} % body
 \put(150,150){\vector(-1,-6){15}} % left arm
 \put(150,150){\vector(2,-1){50}} % right arm
 \put(150,80){\line(-1,-1){50}} % left leg
 \put(150,80){\line(1,-7){15}} % right leg
\end{picture}
```

Figure 2. *This example exceeds standard LaTeX's capabilities. Slopes specified for the left arm and right leg are too steep.*

Figure 2 demonstrates these problems by drawing the same picture with different slopes and diameters. No picture was produced when this was run under standard LaTeX because first the "−6" for the left arm exceeded ±4, caused a LaTeX error message, and halted execution; then the "−7" for the right leg exceeded ±6 and LaTeX stopped again.

When these two simple problems were "corrected" by substituting "$(-1,-4)$" for the left arm and "$(1,-6)$" for the right leg, the picture still did not look as desired — the head was 40pt in diameter instead of 70pt. This messed up the body as well, causing the head to "hover" above the body. In

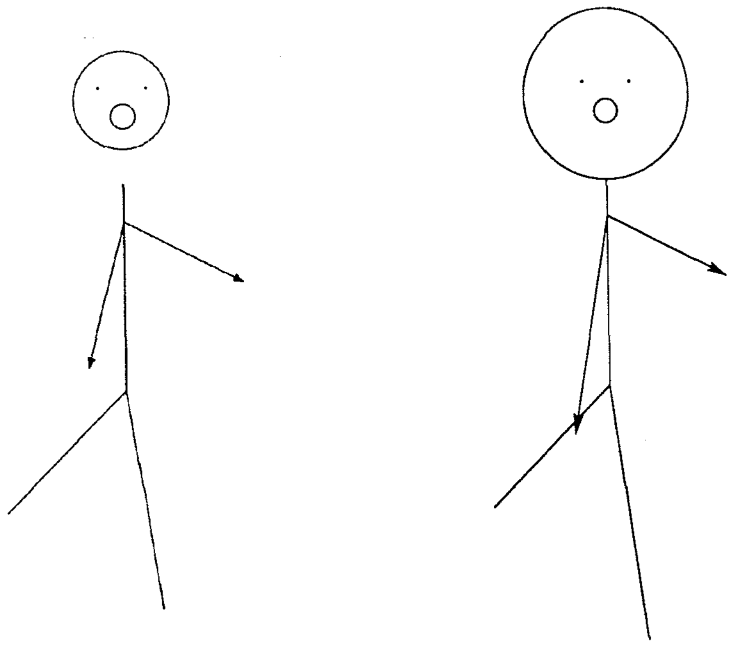


Figure 3.
The left picture was produced by standard LaTeX, the right picture by extended LaTeX.

Figure 3, the left picture is how standard LaTeX did it. On the right is the Extended LaTeX version of the original example.

The LaTeX Picture Environment Extensions

The PostScript printer language is very powerful. It has a very general and flexible set of capabilities enabling the creation of practically any textual or graphic image. It can draw letters, lines, and circles of virtually any size and slope. It can shade the inside of any closed figure it draws. And it allows lines of text to be drawn at any angle. The extended version of LaTeX uses these PostScript capabilities to provide a number of features not normally possible with LaTeX. All extensions are modelled closely after the way LaTeX normally does things.

Extensions to Existing LaTeX Commands

Line Thickness Standard LaTeX provides two line thicknesses which are selected by typing either `\thinlines` or `\thicklines` inside the `picture` environment. These two declarations simply tell TeX which font to use, either `line` or `linew`, respectively. LaTeX version 2.05 also provided a `\linethickness{}` command with which the user could vary the thickness of *non-oblique* (vertical or horizontal) lines.

The latest release of LaTeX does not document the use of the `\linethickness{}` command, probably because of the confusion which arose because it could only do non-oblique lines. With the LaTeX extensions, though, it has been brought back. By typing something such as `\linethickness{5pt}` you can set the line thickness of a PostScript-drawn line or circle to 5pt.

`\thinlines` is equivalent to the command `\linethickness{0.4pt}`. `\thicklines` is equivalent to typing `\linethickness{2pt}`. The Extended LaTeX version of `\thicklines` draws a much thicker line than regular LaTeX `\thicklines` does. Lines of intermediate thickness may be drawn with the `\linethickness` command.

Circles Circles of practically any diameter may be drawn. For hollow circles, use `\circle{`*diam*`}`. For solid circles, use `\circle*{`*diam*`}`.

Lines and Vectors Length and accuracy are virtually unlimited. Line lengths and \put coordinates need not be restricted to integer values as is the case normally with LaTeX. However, whole numbers are still required for Δx and Δy. This might seem restrictive, but if you want a line 34.5 points long with a slope of, say, 0.66, try \line (100,66){34.5}. A negative length causes the line to project opposite to the specified direction.

The \vector command uses the same syntax as the \line command and is just as versatile. The arrowhead points away from the location of the \put command. If you choose to use a thick line, you'll find the arrowhead may not look very good since it was designed to be used with thinner lines. In this case, use the \PSarrowhead command (described later) to custom-make your own vector with a bigger arrowhead.

New LaTeX Commands

PSoval If you've used LaTeX's picture environment, you have seen that the \oval command doesn't really produce an oval at all; it's actually a rectangle with rounded corners. The corners are made up of the same fonts that make circles. Because \oval produces this unique-looking figure, the original LaTeX macro was not redefined. Instead, a new command, \PSoval, was created. The syntax of \PSoval is as follows:

\PSoval{*width*}{*height*}

The \put command specifies the location of the center of the ellipse. Again, width and height are virtually unlimited.

You can get a solid oval by typing \PSoval* with the same syntax as \PSoval.

Two drawbacks — you can't use \PSoval to put text inside the oval. Use an extra \put command to do that if you need to. Also, this release of \PSoval has no provision for printing a portion of an oval — something LaTeX *can* do with \oval.

PSarrowhead Extended LaTeX allows the user to control the shape of arrowheads. The user can make almost any size or shape of arrowhead desired. Add a line and get a custom-made vector as well. The syntax of the command is:

\PSarrowhead(Δx, Δy){*length*}{*width*}{*depth*}

where *length*, *width* and *depth* are defined in the summary. Keep Δx and Δy as whole numbers as in the \line and \vector commands. You can use almost any values you want for the length, width and depth. Use \PSarrowhead* with parameters as above to get a solid arrowhead.

The Extended LaTeX \vector command uses a solid arrowhead that has a length of 8pt, a width of 4pt, and a depth of 2pt.

PStilt Extended LaTeX, because it can use PostScript capabilities, allows the user to temporarily "rotate" the coordinate system of the page for individual lines. This means that text no longer has to be horizontal. In fact, it can be at any slope desired. The syntax of \PStilt is

\PStilt(Δx, Δy){*object or text*}

The user can put text or a picture object like a \framebox inside the third (brace enclosed) argument to \PStilt. This addition is very useful for labelling lines and rotating LaTeX picture objects such as \oval's and \framebox'es.

PSpath Extended LaTeX allows a much easier way of drawing straight lines — by naming the coordinates of the endpoints. In fact, a series of connected lines can be drawn with just one command. The syntax of \PSpath is

\PSpath(x_0, y_0){(x_1, y_1)(x_2, y_2)$\cdots$(x_n, y_n)}

You don't even need the \put command, since the starting point is defined to be (x_0, y_0) and the n points are then connected in sequence. The number of points that can be connected using just one \PSpath varies from system to system. If you try to use a lot of points and get "capacity exceeded" errors, use more \PSpath commands.

The map example of Figure 6 demonstrates a series of points connected as one \PSpath.

Conclusion

The LaTeX extensions described in this article represent one way that TeX and PostScript can be used in a combination that is more powerful than either one alone. While the implementation described here is available only with Textset's DVILASER/PS program, other individuals or organizations might make use of a similar strategy to design their own individualized LaTeX extensions.

Note added in press

Leslie Lamport read a preprint of this article and had several much appreciated comments. He correctly guessed that we were not, at the time the article was written, aware of the new option that draws quadratic Bezier splines. He suggested for portability's sake that the extensions be enabled with a document-style option; the extensions in fact already must be enabled by a command and will work within any document style. We incorrectly stated that the \linethickness command was

Figure 4.

Summary of Syntax of LaTeX Extensions.

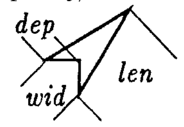
Control Macros

`\PSextensionsOn`	Enables LaTeX extensions.
`\PSextensionsOff`	Disables LaTeX extensions and reverts to regular LaTeX processing.

Redefined Macros

`\linethickness{`*dimen*`}`	Sets the thickness of all lines, including sloped lines, circles, and other geometric shapes. *dimen* is something such as `0.01in` or `3pt`.
`\thinlines`	Sets line thickness to 0.4pt.
`\thicklines`	Sets line thickness to 2pt.
`\circle{`*diam*`}`	Puts a hollow circle at the current location with diameter *diam*, which can be to almost any size and accuracy desired.
`\circle*{`*diam*`}`	Just like `\circle`, only the circle is solid.
`\line(`Δx,Δy`){`*length*`}`	Puts a line starting at the current point with a slope $\Delta y/\Delta x$ and a length defined as described in Lamport's book, where *length* is how far *over* the line is to go in terms of x-units, unless the line is vertical, where *length* then means the length of the line up or down. Use whole numbers for Δx and Δy. *length* can be almost any number to virtually any accuracy.
`\vector(`Δx, Δy`){`*length*`}`	Just like `\line` above, except in addition, a solid arrowhead is added at the other end of the line, pointing away from the `\put` point.

New Macros

`\PSarrowhead(`Δx, Δy`){`*length*`}{`*width*`}{`*depth*`}`	Puts a hollow arrowhead with its tip at the position of the `\put` command, pointing at a slope $\Delta y/\Delta x$ and where the arguments are as shown:  When *depth* is positive, the figure produced is a concave arrowhead as shown. But when *depth* is negative, the figure is convex, and other shapes can then be created, like diamonds and rotated squares.
`\PSarrowhead*{`*same as above*`}`	Puts a solid arrowhead at the position specified by the `\put` command.
`\PSoval{`*width*`}{`*height*`}`	Puts a hollow ellipse with its center at the position specified by `\put`.
`\PSoval*{`*width*`}{`*height*`}`	Puts a solid ellipse with its center at the position specified by `\put`.
`\PStilt(`Δx, Δy`){`*object*`}`	Puts *object* at the position specified by `\put`, but at a slope $\Delta y/\Delta x$. *object* can be text, or any `picture` environment object such as `\oval`, `\PSarrowhead`, or `\framebox` with their appropriate arguments included.
`\PSpath(`x_0, y_0`){(`x_1, y_1`)`$\cdots$`(`x_n, y_n`)}`	Connects the points (x_i, y_i) with straight lines starting at the point (x_0, y_0) and ending with (x_n, y_n). Can have an arbitrary number of points, but the exact number varies from system to system.

no longer documented; see page 199 of the LaTeX manual. He pointed out that our "oval" shape is actually an ellipse, while his, though not an oval, is a convenient shape in which to insert text. Textset plans to implement several other of his suggestions into the program.

```
\setlength{\unitlength}{0.125in}
\begin{picture}(56,80)(11,-12)
\PSextensionsOn
\put(0,77){\framebox(31,3){\huge
  Directions to \bf TEXTSET}}
\put(34.5,61){\huge $\star$}
\put(31.5,58){\vector(1,1){3}}
\put(27.7,57){\bf TEXTSET}
\put(27.3,56){\sf P.O.~Box 7993}
\put(27.3,55){\sf 416 Fourth St.}
\put(25.4,54){\sf Ann Arbor, Mi. 48107}
\put(27,53){\sf (313) 996-3566}
\put(25,48){\Huge Ann Arbor}
\linethickness{5pt}
\PSpath(0,74){(3,72)(5,68)(5,40)(40,3)(56,3)}%I-94
\thicklines% Jackson/Huron
\PSpath(0,71){(10,67)(24,69)(40,68)(56,68)}
\PSpath(17,72){(24,69)}  % Dexter
\PSpath(9,72){(9,63)(24,42)(46,42)(56,38)}% Stadium
\PSpath(13.3,57.1){(41,65)(51,65)}  % Liberty
\put(51,0){\line(0,1){75}}  % State
\PSpath(21,11){(41,31)(41,52)(45,80)}  % Main St.
\put(42.5,61,8){\line(12,-12){13.6}}  % Packard
\put(41.4,54.5){\line(1,0){14.5}}  % Hill
\put(36,62){\line(1,0){15}}  % William
\put(36,63.7){\line(0,-1){7}}  % Fourth St.
\put(51,63.5){\line(1,0){5}}  % N. University
\thinlines \put(56,23){\line(-23,57){23}}% RR Track
\multiput(54.7,25.5)(-0.5,1.24){44}%
{\line(57,23){0.5}}% Ties for RR Track
\put(44,5){\framebox(6,4){\parbox{0.6in}%
  {\sf Briarwood\\Mall}}}
\put(5,70){\framebox(3,4){\sf Plaza}}
\put(34,37){\framebox(6,4){\parbox{0.4in}%
  {\sf Pioneer High\\School}}}
\put(51.5,58){\framebox(4,4){\parbox{0.4in}%
  {\sf Univ of Mich}}}
\thicklines
\put(11,16){\circle{2}}  % Ann Arbor
\put(17.5,14){\circle{4}}  % Detroit
\put(13,8){\circle{3}}  % Toledo
\put(4,4){\framebox(16,16){}}  % Inset Frame
\linethickness{5pt}
\put(42.5,44.5){\PSoval{2}{3}}  % Michigan Stadium
\put(44,46){\sf Michigan} \put(44,45){\sf Stadium}
\thinlines
% Inset to follow
\PSpath(18,14){(10,14)(9,16)}  % I-94
\put(9,16){\vector(-1,0){4}}  % I-94
\put(13,7){\line(0,1){13}}  % US 23
\put(17,7){\line(-1,0){13}}  % Ohio Turnpike
\put(17,7){\vector(1,-1){2}}  % Ohio Turnpike
\PSpath(10,4){(16,10)(20,18)}  % I-75
\thinlines  % Ohio-Michigan Border
\multiput(4,9.5)(0.5,0){32}{\line(1,0){0.25}}
\put(1,59){\framebox(3,2){\large \bf I-94}}
\put(53,4){\framebox(3,2){\large \bf I-94}}
\put(45,1){\large \it Exit 177}
\put(29,17){\large \it Exit 175}
\put(-1,67){\large \it Exit 172}
% street names to follow
\put(28,19){\PStilt(1,1){\small \sf
  ANN ARBOR - SALINE ROAD}}
\put(27,42.4){\small \sf STADIUM BLVD}
\put(8,63){\PStilt(0,1){\small \sf STADIUM}}
\put(14,66.5){\PStilt(14,2){\small \sf JACKSON}}
\put(20,71){\PStilt(7,-3){\small \sf DEXTER}}
\put(29,69.4){\PStilt(16,-1){\small \sf HURON AVE}}
\put(29,62.4){\PStilt(28,8){\small \sf LIBERTY ST}}
\put(37,56){\PStilt(0,1){\small \sf FOURTH ST}}
\put(44,62.5){\small \sf WILLIAM}
\put(52,64){\small \sf N UNIV}
\put(52,55){\small \sf HILL ST}
\put(46.3,58.4){\PStilt(1,-1){\small \sf PACKARD}}
\put(54.3,50.4){\PStilt(1,-1){\small \sf ROAD}}
\put(50,23){\PStilt(0,1){\small \sf STATE}}
\put(50,43){\PStilt(0,1){\small \sf STREET}}
\put(40.5,46){\PStilt(0,1){\small \sf MAIN}}
\put(43,70){\PStilt(4,28){\small \sf STREET}}
\put(5,18){\bf ANN} \put(5,17){\bf ARBOR}
\put(5,15){\small \it Chicago}
\put(5,14){\small \it 225 Mi}
\put(10,13){\small \sf I-94}
\put(12.5,16.5){\large \bf DETROIT}
\put(10.2,11){\small \sf US 23}
\put(16.5,10){\small \sf I-75}
\put(4.4,6){\small \sf OHIO TURNPIKE}
\put(11,4.5){\small \it Cleveland 100 Mi}
\put(14.5,8){\bf TOLEDO}
\put(6,9.8){\tiny Michigan}
\put(6.8,8.8){\tiny Ohio}
\put(52,1.8){\it To Detroit}
\put(0.5,74.5){\it To Chicago}
\put(14.5,13){\PSarrowhead(0,1){0.5}{1}{0.35}}
\put(14.45,12.2){\framebox(0.1,1){}}
\put(13.3,11.4){\tiny Detroit}
\put(13.5,10.8){\tiny Metro}
\end{picture}
```

Figure 5. *This extended LaTeX code and the DVILASER/PS system produced the map in Figure 6 (facing page).*

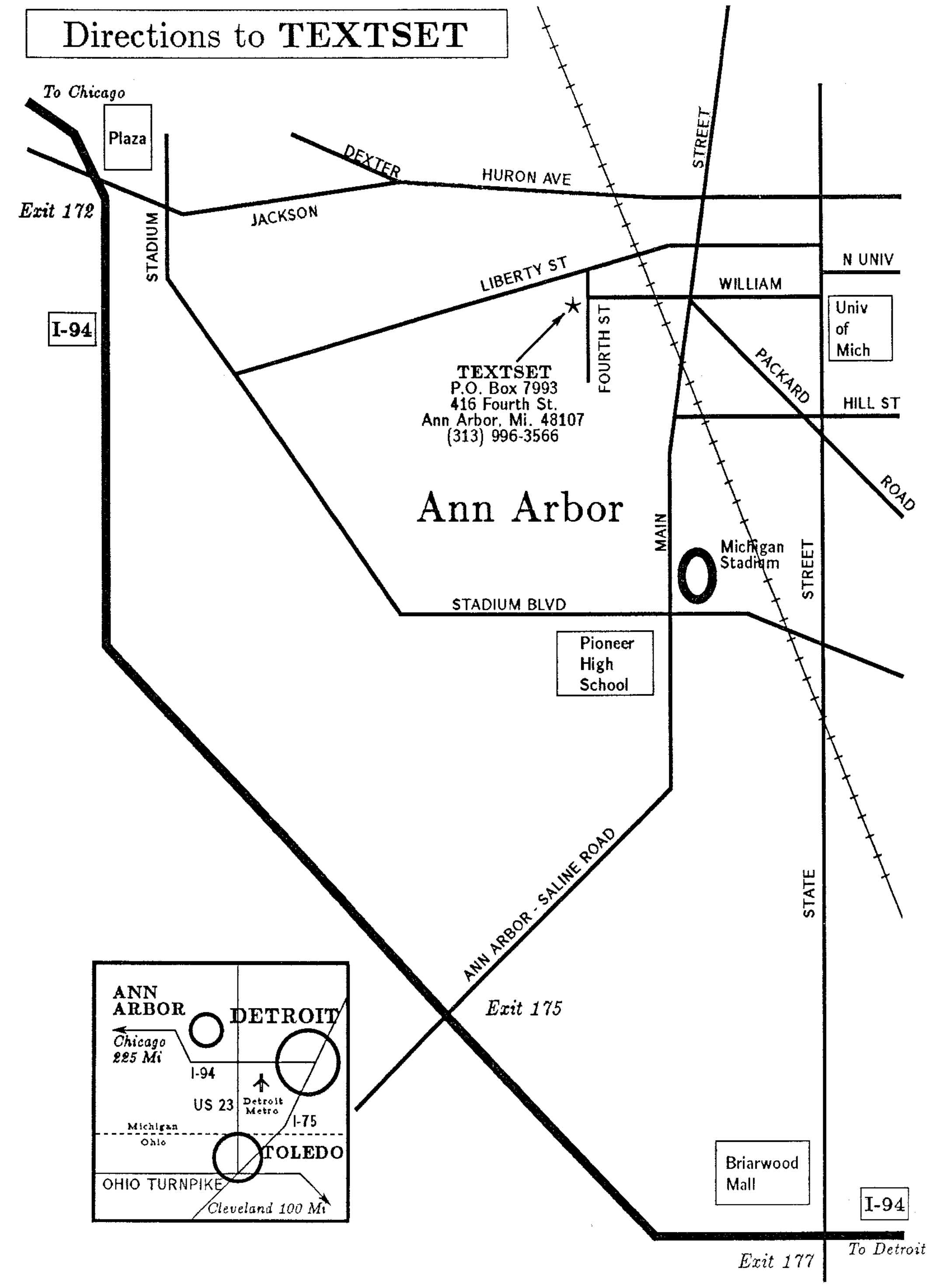

Directions to TEXTSET
To Chicago
Plaza
Exit 172
I-94
STADIUM
JACKSON
DEXTER
HURON AVE
STREET
LIBERTY ST
N UNIV
WILLIAM
Univ of Mich
FOURTH ST
PACKARD
ROAD
HILL ST
TEXTSET
P.O. Box 7993
416 Fourth St.
Ann Arbor, Mi. 48107
(313) 996-3566
Ann Arbor
MAIN
Michigan Stadium
STREET
STADIUM BLVD
Pioneer High School
ANN ARBOR - SALINE ROAD
STATE
Exit 175
ANN ARBOR
DETROIT
Chicago 225 Mi
I-94
US 23
Detroit Metro
I-75
Michigan
Ohio
TOLEDO
OHIO TURNPIKE
Cleveland 100 Mi
Briarwood Mall
I-94
To Detroit
Exit 177

Figure 6.

Results of the 1985 TUG Questionnaire

Pat Fina

Response to the 1985 TUG Questionnaire was better than anticipated; 187 questionnaires, including three composite responses, were received. The following summary should be read with caution because (1) there was a wide range in the number of responses to individual questions, from near 100% on some to a low of 24 replies to question #21 (needs of METAFONT users not met by TUG); (2) categorizing responses to the essay-type questions is necessarily a subjective task; (3) some individuals replying to questions #3, 6, 8 and 19 chose not to order their responses from 1–*n* as requested; and (4) the scorer's ignorance of acronyms and abbreviations common in science undoubtedly caused some error in compiling the results for questions on professional journals, associations, TEX competitors and the like.

TEX User Profile

Of those responding, 36% first learned about TEX at their workplace, 27% from a friend, 22% through familiarity with the works of Donald Knuth, 4% from journal articles, 2% at conferences, and 9% from other sources. 3% have never actually used TEX; 31% have used it less than one year; 24%, two-three years; 10%, three-four years; 5%, four-five years; and 5%, more than five years. 3% classed themselves as non-users; 22%, beginners; 53%, intermediate users; 22%, advanced users; and a courageous 4%, true wizards. 33% gave their principal occupation as computer programmer, systems manager or systems operator; 24%, researcher; 19%, teacher; 6%, secretary; and 18%, other. That only 3% of the others listed themselves as students perhaps reflects that the questionnaire was distributed during the summer, or perhaps raises the question whether TUG should offer a discount student membership rate.

Regarding educational resources used in learning TEX, 25% relied on *The TEXbook*, 13% on *TEX and METAFONT*, 12% on *TUGBOAT*, 10% on friends, 8% on *First Grade TEX*, 8% on local documentation, 7% on *The LATEX Manual*, and 17% on all other sources. For the reasons noted above, the validity of any rank ordering of the importance of these sources is suspect; however, *The TEXbook*, *TEX and METAFONT*, and TUG courses were the three options most often ranked as most important or useful by those following the 1–*n* numbering system. The responses to question #4 (educational resources you lacked when learning TEX) were varied: 21% focused upon humans (gurus, teachers, an active local TEX community, etc.) in their answers; 8% noted hardware or software problems (no working version of TEX, inadequate output devices and the like) as their primary impediment in learning TEX, and 7% listed miscellaneous causes such as lack of time. A solid 64% majority of responses, however, concentrated upon inadequate documentation, with 12% desiring a master reference manual of TEX commands; 10%, a beginner's manual; 10%, sample input files and templates; 7%, technical memos (about drivers, font files, interfacing to operating systems, etc.); and 25%, various miscellaneous forms of documentation.

TEX is being used to typeset many kinds of documents: 22%, letters and memos; 21%, scientific reports; 16%, software documentation; 12%, overhead projector transparencies; and 29%, all other kinds. TEX was rated as being most important or useful for scientific reports and least useful in typesetting tables of data, an unsurprising response. Only 3% of the respondents indicated they use `plain.tex` *au naturel*; 38% combine it with macros of their own, 11% with a personally designed macro package, 15% use LATEX, 12% use macros developed by their employer, and 21% use other macro packages.

Judging from the replies to questions #9–15, the current TEX user population is drawn primarily from the fields of computer science, electrical engineering, mathematics, physics and publishing. Those responding noted they subscribe to or read regularly 177 journals: 19%, *CACM* and other ACM publications; 16%, IEEE publications; 37%, other computer journals; 10%, math journals; 7%, physics journals; and 11%, all others. Respondents thought slightly less than half of these journals would welcome articles about TEX, with *Byte*, IEEE and ACM publications mentioned most often, and with a *New Yorker* profile of Donald Knuth being perhaps the most creative suggestion of the lot. Respondents indicated membership in 72 different professional associations: ACM, 24%; IEEE, 15%; AMS, 6%; DECUS, 5%; MAA, 5%; and all others, 40%. Seventy different conferences and meetings were listed as having been attended in the past year, with DECUS, TUG and SIGGRAPH being the only three listed ≥5% of the time. 25% of the respondents are willing to volunteer to write a journal article about TEX, 15% are willing to advocate its adoption by professional associations they belong to, and 21% are willing to give talks about TEX at conferences.

25% of the respondents to question #16 indicated they have access to the ARPANET, 13% to USENET, 12% to BITNET, 9% to CSNET, 8% to UUCP and 33% to other computer networks and bulletin boards.

METAFONT User Profile

Of those responding, 85% have never attempted to use METAFONT; 7% have used it less than one year; 1%, one-two years; 4%, two-three years; 2%, three-four years; and 1%, four-five years. 76% of those replying to question #18 classed themselves as non-users; 18%, beginner; 3%, intermediate user; 2%, advanced user; and 1%, true wizard. 38% of users learned METAFONT from *TEX and METAFONT*, 20% from *The METAFONTbook*, 13% from friends, 12% each from *TUGBOAT* and local documentation, and 5% from all other sources. Nearly half (49%) of METAFONT users reported lacking a working version of the program or had similar software problems when learning METAFONT; with 28% noting need for better documentation; 7%, lack of teachers and other users; and 16%, miscellaneous problems.

TEX Site Profile

85% of those responding reported that TEX is in use or being installed at their place of employment. 40% work for non-profit educational institutions; 25%, non-profit research institutions; 30%, for-profit corporations; and 5%, others. TEX is supported and new users trained by paid staff at 40% of TEX sites, by volunteers or students at 27% of the sites, and is unsupported at 33% of employers using TEX. 41% of respondents estimated there are 0–10 users at their site; 32%, 11–25 users; 14%, 26–50 users; and 13%, more than fifty users. 66 different typesetting and text formatting programs were listed as TEX's "competition": Troff, 19%; Runoff, 17%; Script, 10%; trix red/redpp, 9%; nroff/dtroff, 7%; Scribe, 5%; and all others, 33%.

TUG Services

80% of the respondents consult the membership list occasionally; 12%, frequently; and 8%, never. 26% thought the list is fine in its current format, 19% asked for more sorting and indexing (by geographic location, wishful vs. real users, etc.) or to have on-line access to the list to produce individualized sorts, 43% wanted more fields of data (or current fields like electronic mail addresses and applications completed more often), and 12% requested that changes and corrections sent to TUG be updated in subsequent lists.

38% have consulted a TEX site coordinator and rated the response they received as 2.16 on a scale from 1 = excellent to 5 = poor. Of those who have contacted on-line information sources about TEX, 54% consulted TEXhax, 33% consulted Laser-lovers, and 13% used other sources. Average scores for quality of response (on the 1–5 scale) were: 2.29 for TEXhax, 2.17 for Laser-lovers and 2.12 for other sources.

TUGBOAT subscribers reported that 19% read technically-oriented articles; 9%, the administrative news; 70%, both; and 2%, neither. Suggestions for future articles were difficult to categorize: 42% wanted macro-related topics (descriptions of LATEX and other widely used packages, sample solutions to common formatting problems and the like), 30% wanted more information about related software like on-screen "preview" programs or about how TEX compares with other text formatting software, 15% wanted regular news about METAFONT, and 6% wanted people-related information (job opportunities, listings of products and services offered by paid TEX consultants and TUG volunteers, etc.). Of the 63 individuals willing to volunteer for

TUGBOAT assignments, 14% would guest edit an issue; 21%, edit a column; 64%, write an article; and 1%, other.

TUG Meetings

63% of those responding have never attended a TUG meeting; 19% have attended one; 12%, two; 4%, three; 1%, four; and 1%, five. Travel expenses (39%) were the most frequently listed reason for not attending, followed by conflicts in schedule (20%), lack of interest (15%), high meeting fees (9%), and other reasons (17%). Suggestions for locations of future meetings were: New England/New York City, 28%; East Coast, 17%; deep South (east of Texas), 4%; Midwest, 5%; Colorado, 3%; Texas, 6%; Southwest (east of California), 2%; Pacific Northwest, 5%; northern California, 5%; southern California, 4%; Europe, 11%; Canada, 4%; Australia, 1%; and alternating locations, 5%. The Steering Committee has already taken action on this item by scheduling the 1986 TUG meeting for Tufts University in suburban Boston. 19% of respondents were willing to act as hosts for future TUG meetings in their areas.

Concerning regional TEX speaker series, 20% of the respondents are willing to organize such events, and 83% of respondents indicated they would probably attend, being willing to travel an average 73.5 miles for an evening or 151.7 miles for a weekend event. 5% thought such meetings should be scheduled monthly; 37%, quarterly; and 58%, on an ad hoc basis.

TUG Courses

71% of the respondents have never attended a TUG-sponsored course; 15% have taken one; 11%, two; 2%, three; and 1%, four or more. Reasons for not attending courses closely paralleled those for not attending meetings: travel costs (40%), lack of interest (18%), high course fees (17%), and other reasons (25%). 69% thought beginning TEX should be taught using `plain.tex`; 17%, LATEX; other macro packages, 8%; and a choice of macro packages or `plain.tex`, 6%.

Responses to question #47 about course fees averaged to $404 for a 5-day beginning course, $269 for a 3-day intermediate course and $190 for a 2-day advanced seminar. Some individuals expressed a desire to have course fees charged by class rather than per student, and others indicated the courses should be priced at cost. Maximum desirable enrollment for courses with lab sessions averaged to 20 students and for lecture-only courses to 47 students; 40% think that labs are necessary at the beginning level, 33% at the intermediate level, and 27% for certain advanced courses. Topics suggested for advanced seminars correspond closely with the current TUG curriculum: macro writing and other macro-related topics (53%), METAFONT, typography and book design (15%), output routines (8%), internal workings of TEX82 (9%), hardware-specific seminars (10%), and miscellaneous (5%).

Summary

In general, results of the 1985 TUG Questionnaire presented few surprises; the need for more documentation and for greater interaction among TEX users have been pressing concerns of the TUG leadership for years. Many of the essay-type responses touched upon controversial topics, such as the individual who refused to answer questions #10–15 (about journals, professional associations and conferences that might welcome information on TEX) with the comment: "Let the people who profit from TEX promote its use... TEX is not a religion!" Rest assured that TUG has no plans for the deification of Prof. Knuth; rather, those questions are meant to reflect that TUG's charter is that of a volunteer organization, viable only through the unpaid efforts of its members. *Everyone* who uses TEX profits from the fact that the program is in the public domain. Fewer than one per cent of those responding to the questionnaire indicated they consider TEX consulting their primary occupation, and until the user population increases to the point where specialists *can* earn a living solely from their TEXpertise, the further documentation and macro packages requested by average users simply will not be developed. Perhaps the greatest benefit of the TUG questionnaire was in eliciting the names of 59 volunteers for various project assignments; special thanks to those willing to assume such responsibilities.

A final note. The author has made a resolution never again to produce a non-machine-scorable questionnaire of this length and complexity.

TEX TEXT EDITORS FOR THE IBM PC

A. Hoenig
M. Pfeffer

What's the best way to get material into your computer for use by TEX? The question was moot in the pre-PC era of TEX; you simply made do with whatever editor was available on your system. Editors and word processors have become big business in the PC world, some being strikingly better-designed than others, and the one you use can drastically affect your efficiency. Recently, I looked at a few text editors available for the IBM PC (and compatibles).

What are some of the characteristics a perfect text editor should possess? Remember, the sole purpose of any such editor is to accept your text into a source file that TEX can read and compile. Formatting by the text editor is a minor concern, since TEX handles all that for you.

The editor should produce standard ASCII output. Right away, this disqualifies most of your high-flying, superstar word processors—your WordStars, Multimates, Microsoft WORDs, and so on—which depend upon the presence of nonprintable ASCII characters in the text file to control format characteristics. But the ideal editor should be capable of elementary word- and text-processing capabilities—word wrap (so you don't have to hit the carriage return key at the end of each line), block management (so you can shuffle and reshuffle large chunks of text quickly and easily), and search and replace capabilities.

It should also work *fast*. Surprisingly, most of the superstars aren't, slowly saving files and searching for text strings as if being paid by the hour. I guess the business marketplace puts no premium on speedy execution (where many office workers *are* paid by the hour). This is unacceptable in a TEX session, when you're typically in a frenzy to correct that one last mistake and recompile your source file.

Finally, the editor should *not* be copy-protected. Whatever the relative merits of copy protection for the software developer, it's definitely a drag for the software user.

In addition, there are several other purely idiosyncratic conventions to which I expect my editor to conform. Most controversially, I want to to be able to use a mouse with it. A *mouse* is a palm-sized, electronic gadget, vaguely reminiscent of a real mouse, which you roll around your desk. These rolling motions translate into movement of the on-screen cursor. Pressing a button on the mouse usually affords additional editing control. Controversy arises because many power users find that any productivity gains are offset by the effort and time you waste moving your hand from keyboard to mouse. Admittedly, the mouse is irrelevant during the initial text entry operation, but it's worth its weight in Velveeta during the rewrite process, when keyboard control is less crucial. In my case, the time spent on rewrite dwarfs the time spent on the initial keyboard entry, so my mouse plays a crucial *roll* in productivity enhancement.

Next, I expect to be able to configure my keyboard so that the various shift keys are *sticky*, that is, when you press them they stay in effect for the duration of the next keystroke. The movement towards sticky-shift operations arose in response to the complaints of physically-handicapped users (who may not be able to press, say, the Ctrl and C keys at the same time), but I've found it to be the greatest thing since sliced bread, though I'm blessed with no handicaps (at least no physical ones). With a sticky-shift editor, typing an uppercase "A" (for example) becomes a sequence of 2 strokes, rather than a pair of simultaneous strokes. Your typing rhythm is less disturbed when you keep to a linear succession of single strokes. Nevertheless, some editors are written to hog the keyboard interrupts, and won't let you "stickify" the keyboard.

In this column, I report on my usage of 3 text editors which distinguish themselves in various ways. All produce ASCII text files, are *not* copy protected, and do their jobs fast enough to be at or close to the limits of human perception.

PC-Write

PC-Write is a nifty little program. Although it was manufactured as a full-featured word processor (which it in fact is), it functions beautifully as a text editor. Like most current-generation word processors, PC-Write works by accepting as commands non-standard key sequences — that is, key sequences involving the function keys, the Escape key, or the Alt or Control shift keys.

The thing I like best about PC-Write is its customizability. Keys can be reprogrammed so that single keystrokes represent dozens of PC-Writecommands, or you can change the command sequences from the PC-Write standard to your own standard. On the fly, you can define a single keyboard "macro," but a PC-Write macro is simply a concatenation of different commands and, as such, possesses a fraction of the sophistication of a TEX macro. Sticky shifts are one of the things you can customize into your version of PC-Write.

There's a fearful symmetry between TEX and PC-Write — both are in the public domain. A disk with the program and reference manual file (it prints out to 180 pages) can be yours for a whopping 10 bucks. Set the program up, run off a copy of the manual, and you're off and running. For a few more dollars, the source listing of the program (a mix of assembly language and Pascal) is yours. The developers, Quicksoft, Inc., are pioneers in *freeware* — they hope its use will spread by word of mouth, and they rely on your sense of fair play to remit an additional contribution to them.

PC-Write comes with support for several of the standard mice. Also, the editor lets you configure it so that shift keys are sticky. (It's a simple matter of changing a special `ruler.def` auxiliary file to which PC-Write is accustomed to check for ruler-related matters and customizations.)

PC-Write, although good, is not perfect. For one thing, it's limited to document files of 64k or less. If you have larger files, it handles them only with a clumsy swapping maneuver — pieces of your file are swapped back and forth from disk to memory as you need them. Furthermore, PC-Write may not be so convenient for programmers. Although you can indent your programs, you have to find out how on your own. (The manual gives little help.) It can't check your matching delimiters for you.

I reviewed version 2.55, the current version. Reportedly, a new version will be available in February or March, presumably for the same price. This new version should handle larger files, hopefully with no degradation in speed, and should come with a spelling checking program, among other enhancements.

The Norton Editor

Peter Norton, no relation to Ed, has won fame and fortune writing as a PC guru. He writes extensively about the guts of the IBM PC (the source of his fame). He's also marketed a clever bunch of programs which extend the PC's operating system in a few much-needed ways (the source of his fortune). Recently, he's lent his prestige to a cute little editor — The Norton Editor — which he bills as *the* programmer's editor.

The Norton Editor makes no pretense about being a word processor. It's incapable of doing many fancy word processing tasks, but does program text editing well. Like PC-Write, it word wraps, manages blocks, and searches and replaces. NE commands are two-key sequences, the first key being one of the function keys, and the second executing a command. Pressing a function key summons a menu of choices to the bottom of your screen, so you don't have to memorize the command sequences. Fortunately, once you know what commands you want, you can tap the key sequences at high speed, and ignore the menus.

This editor is easy to learn. Once you've mastered a short pamphlet (the entire documentation) you know all there is to know about it.

NE provides no sticky-shift provision within the editor, but it's easy enough to get hold of a public domain sticky-shift program (from a local bulletin board) and run it before invoking the Norton Editor. As with PC-Write, there is built-in mouse support.

NE is superior to PC-Write in its ability to work as a programmer's text editor when you've finished using it for TEX. There are a variety of ways of dealing with program indentation and checking of matching delimiters.

The Norton Editor is a new product, and Norton will probably address some missing features in future versions. For example, NE is bereft of any semblance of a macro facility, a shocking omission.

Like PC-Write, this editor is cheap. It lists for $50, but you can do better at some mail-order houses.

Epsilon

Epsilon is a sophisticated, programmer's text editor closely modelled on the EMACS editor. Epsilon will apparently and happily work with document files of any size. As fast as the other two editors are, this one is even faster.

This current version (version 3.*x*) has been programmed in EEL — Epsilon Extension Language — and anything (within reason) that you don't like about the editor, you can change, by simply rewriting that module. The folks at Lugaru Software — Epsilon's developers — provide EEL source files for Epsilon and an EEL compiler as part of the Epsilon package. EEL is a C-like programming language.

There are a few things which cannot be changed. Epsilon "steals" the keyboard interrupts, so there's no way you can get a mouse to work with it. In the same way, no BBS sticky-shift utility will work, but at least you can reprogram Epsilon to accept sticky (or shifty?) behavior.

On the other hand, Epsilon gives you far greater control over text editing (provided you need neither sticky shift nor mouse). The editor provides an extensive mechanism for creating buffers to store deleted material, additional files, or whatever. Apart from EEL, there's an extensive macro facility, and you can define and use any number of macros on the fly. You may reassign different command strings to Epsilon commands. The Search facility — whereby you search for a particular string in your text file — is *incremental*. Suppose you're looking for the word "Mozart" in your file. Normally, your text editor waits for you to enter the master's name before commencing the search. Epsilon searches as you enter the word. It senses that first "M," and looks for it. When you enter the "o," Epsilon refines its search for the first occurrence of "Mo." When you type "z," it looks for "Moz," and so on. Incremental searches are usually much faster for you. Lugaru has embedded numerous other impressive bells and whistles in their product.

Epsilon is clearly aimed at power users, and nowhere is this more evident than in the manual, which contains almost no examples. Also, their discussion is steeped in the lingo of EMACS, which is a dialect of the usual text editor terminology. The manual takes a bit of getting used to.

All this power has its price. Epsilon weighs in at $195, more than the other two editors, but still a good deal less than much other software on the market. It may be worth checking prices at mail order outfits who cater to hard-core programmers; maybe such places will carry this program at a discount.

And Now for Something Completely Different — Lightning

I hope you use a spell-checking program in conjunction with your editing. There are any number of good programs out there. Just make sure you use one that allows you to add entries to a user dictionary when they're initially flagged as errors. That way, as you use your checking program over a period of time, all special TEX commands enter that dictionary file. After a short while, the program catches not only your prose misspellings, but also your misspellings of TEX commands. I save a lot of time in TEX compilation that way.

Anyway, a new gimmick in the spell check arena is the program Lightning. Not quite a spelling checker, and yet not quite *not* a spelling checker, Lightning works alongside your text editor. It checks your spelling as you enter your text. As soon as you enter a space (or other word delimiter), Lightning checks the word against a dictionary file it has previously loaded into your computer's memory. If the word has no match in this dictionary, you get a gentle beep, to which you may hearken. Note, though, you *cannot* use Lightning in the traditional spell-check manner. It can't search your document file for you in one fell swoop to identify all errors in one pass. You can only catch errors as you make them. You also get to use Lightning in a Thesaurus mode. In this way/fashion/manner, you can spice up your writing/setting forth on paper/authoring/prose/composition/work with synonyms as you write.

The program performs as advertised, is real easy to use, and is cheap. Borland International is a pioneer in the development of good, cheap software. (Remember Turbo Pascal?) The Lightning program is only $99.95. Lightning works well with both PC-Write and the Norton Editor, but not at all with Epsilon, presumably because Epsilon is so greedy of some system interrupts. Borland touts this product as being in the vanguard of a whole series of intelligent software products using the same software engine as does Lightning.

I have some reservations about the utility of the program. So often, what you enter at the keyboard is just the first draft of many. What's the point of getting it letter perfect, when so much of it will be consigned to that great bit-bucket in the sky? However, if you're the type who has done all your serious revising by hand so the draft you key in is close to final, Lightning could be for you.

I praise the folks at Borland for one aspect of their design. Too often, memory resident programs continue to hog memory long after you're through using them. If you need that memory for some other program, too bad. You can only reclaim this memory by rebooting your system. Not so with Lightning. There is a command which kills the program and knocks it out of the system.

Sources

PC-Write Quicksoft, 219 First N #224, Seattle, WA 98109, (206) 282-0452

The Norton Editor Peter Norton, 2210 Wilshire Boulevard, Santa Monica, CA 90403, (213) 399-3948

Epsilon Lugaru Software, Limited, 5740 Darlington Road, Pittsburgh, PA 15217, (412) 421-5911

Lightning Borland International, Inc., 4585 Scotts Valley Drive, Scotts Valley, CA 95066, (800) 556-2283

TEX Output Previewers

Mark Senn

Sometimes it's faster and cheaper to view TEX output at your desk before or instead of printing it on a laser printer or phototypesetter. A TEX output previewer displays TEX output on a VDU (video display unit).

Previewers are in use on VDUs that have from 80×20 to at least 1152×900 individually addressable elements.

An entire page usually won't fit on a screen when using fonts that are large enough to be read comfortably. By panning (scrolling) one can look at other parts of the page. One can look at different pages by going through the pages in order, skipping to a page by giving its "real" or TEX (`\count0`) page number, or (sometimes) by searching for a word on a page.

The previewer available from Andrew Trevorrow does the best it can given the features of the VDU used for the display. It supports a wide variety of VDUs of varying intelligence, including the AED 483, AED 512, ANSI, DEC ReGIS, DEC VT100, Visual 500, Visual 550. It is described in more detail in "DVItoVDU: A TEX Page Previewer," page 25.

The other previewers listed below work by putting a pixel-for-pixel facsimile on their VDUs. Three different modes are available while running the Trevorrow previewers: *terse* substitutes standard hardware characters for TEX characters; *box* displays the outline of each character's bounding box (the smallest rectangle containing all black pixels for this character); *full* displays full character bitmaps. (On 80w × 20h displays where each character position represents one pixel you won't see much of your document in *full* mode.)

Figure 1.

Video Display Previewers.

- *C* *EGA, Hercules, or Tecmar card required.*
- *D* *Included on the U WASH distribution tape.*
- *E* *Integrated editor allows TEX'ing part of an input file.*
- *H* *Has on-line help file.*
- *P* *Work in progress or in planning stages.*
- *T* *Has tpic support. (The* pic *program is used with the UNIX* troff *typesetting system to produce line graphics, and the* tpic *program allows pic input to be used with TEX.)*
- *W* *Previewer and another program (say an editor) can both be on-screen in separate windows.*
- *Z* *Preview image can be enlarged for close inspection.*

Apollo Aegis *W, Z*	Bruce Baker 313-996-3566	Textset Box 7993 Ann Arbor, MI 48107
BBN BitGraph UNIX 4.2 BSD *D*	Mark Senn 317-743-0560	107 Digby Road Lafayette, IN 47905-1163
DEC VAXstation VMS, UNIX *P, W, Z*	Bruce Baker 313-996-3566	Textset Box 7993 Ann Arbor, MI 48107
DEC VAX VMS *H, Z*	Andrew Trevorrow 08-228-5984	University of Adelaide Computing Centre Box 498, GPO, Adelaide South Australia 5001, Australia
IBM PC MS-DOS *C, Z*	Toni Formichella 617-944-3700x2640	Addison-Wesley Publishing Educational Media Systems Reading MA 01867
IBM PC MS-DOS *C, Z*	Lance Carnes 415-388-8853	Personal TEX 20 Sunnyside, Suite H Mill Valley CA 94941
IBM PC MS-DOS *C, Z*	Bruce Baker 313-996-3566	Textset Box 7993 Ann Arbor, MI 48107
Integrated Solutions UNIX 4.2 BSD *W, T*	David Benjamin	University of California, Irvine Department of Information and Computer Science Irvine CA 92717
Perq PNX *E, W*	Wolfgang Appelt	Gesellschaft für Mathematik und Datenverarbeitung D-5202 Sankt Augustin Federal Republic of Germany
Sun Microsystems UNIX 4.2 BSD *H, W*	Michael Harrison 415-642-1469	University of California, Berkeley Computer Science Division Berkeley, CA 94720
Sun Microsystems UNIX 4.2 BSD *D, T*	Tim Morgan	University of California, Irvine Department of Information and Computer Science Irvine CA 92717
SUN UNIX 4.2 BSD *H, W, Z*	Bruce Baker 313-996-3566	Textset Box 7993 Ann Arbor, MI 48107

TEX OUTPUT DEVICES

Barbara Beeton
American Mathematical Society

Most of the interfaces listed in these charts are not on the standard distribution tapes. Some are considered proprietary. Information regarding these interfaces should be obtained 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.

Preliminary copies of these charts were distributed to all the listed contacts prior to publication of this issue. Just under half responded to verify the existing information or to make additions or deletions. New information is welcome; send it to Barbara Beeton (address on page 4).

Sources

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

ADAPT Adapt, Inc, Marc Berkowitz, 415-393-9500

ADLD Adelaide University, Australia, Andrew Trevorrow, (08) 228 5984

AMS American Mathematical Society, Ron Whitney, 401-272-9500

A-W Addison-Wesley, Toni Formichella, 617-944-3700, ext. 2640

BOCHUM Ruhr Universität Bochum, Norbert Schwarz, 49 234 700-4014 (NOS 2; NOS/VE in progress)

CALTECH Cal Tech, Glen Gribble, 818-356-6988

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

CLMBIA Columbia University, Frank da Cruz, 212-280-5126

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

COS COS Information, Gilbert Gingras, 5647, rue Ferrier, Montréal, Québec H4P 1N1, 514-738-2191 (Jan '86)

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

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

GA TECH GA Technologies, Phil Andrews, 619-455-4583

GMD Gesellschaft für Mathematik und Datenverarbeitung, Dr. Wolfgang Appelt, D-5202 Sankt Augustin, Federal Republic of Germany

HP Hewlett-Packard, Stuart Beatty, 303-226-3800, ext. 2067

IAM Institut für Angewandte Math, Univ of Bonn, Federal Republic of Germany, Bernd Schulze, 0228-733427

IMAGEN Imagen Corp, Dan Curtis, 408-986-9400

INFN INFN/CNAF, Bologna, Italy, Maria Luisa Luvisetto, 051-498286

INTGRPH Intergraph, Mike Cunningham, 205-772-2000

JDJW JDJ Wordware, John D. Johnson, 415-965-3245

K&S Kellerman & Smith, Barry Smith, 503-222-4234

LLL Lawrence Livermore Laboratory

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

MR Math Reviews, Patrick Ion, 313-996-5273

OCLC OCLC, Tom Hickey, 616-764-6075

OSU1 Ohio State University,
John Crawford, 614-422-1741

OSU2 Ohio State University, John Gourlay,
614-422-6653

PERS Personal TEX, Inc., Lance Carnes, 415-388-8853

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

RECAU Aarhus University, Regional Computer Center

SARA Stichting Acad Rechenzentrum Amsterdam,
Han Noot, Stichting Math Centrum,
Tweede Boerhaavestraat 49, 1091 AL Amsterdam
(*TUGBOAT* 5, no. 1)

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

SCI AP Science Applications, L. E. Fields,
619-458-2616

SLAC Stanford Linear Accelerator Center,
Alan Spragens, 415-854-3300, ext. 2849

SRI SRI International

STNFD Stanford University

SUN Sun, Inc

TALARIS Talaris, Sonny Burkett, 619-587-0787

T A&M1 Texas A&M, Bart Childs, 409-845-5470

T A&M2 Texas A&M, Ken Marsh, 409-845-4940

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

TEXET TEXET Corp, Lance Carnes, 415-388-8853

TEXTSET Textset Corp, Bruce Baker, 313-996-3566

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

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

UCIRV1 University of California, Irvine,
David Benjamin

UCIRV2 University of California, Irvine,
Tim Morgan

U DEL University of Delaware,
Daniel Grim, 302-451-1990

U KOLN Univ of Köln, Federal Republic of Germany,
Jochen Roderburg, 0221-/478-5372

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

U MD University of Maryland, Chris Torek,
301-454-7690

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

U MILAN1 Università Degli Studi Milan, Italy,
Dario Lucarella, 02/23.62.441 (329)

U MILAN2 Università Degli Studi Milan, Italy,
Giovanni Canzii, 02/23.52.93

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

U WASH1 University of Washington,
Pierre MacKay, 206-543-2386

U WASH2 University of Washington,
Jim Fox, 206-543-4320
(NOS 2.2)

U WISC University of Wisconsin,
William Kelly, 608-262-9501

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

WASH ST Washington State University,
Dean Guenther, 509-335-0411

WZMN Weizmann Institute, Rehovot, Israel,
Malka Cymbalista, 08-482443

YALE Yale University, Bill Gropp, 203-436-3761

Figure 1.

Laser Xerographic, Electro-Erosion Printers

[1] *graphics supported*

[2] *included on one of the standard distribution tapes*

	Agfa P400	Apple LaserWriter	Canon	Corona Laser Printer 300	DEC LN01
Amdahl (MTS)					
Apollo		TEXTSET[1]			
Apple Macintosh		K&S[1] A-W[1]			
Cadmus 9200					
CDC Cyber					
DECSYSTEM 10					
DECSYSTEM 20					
DEC VAX UNIX		CRLTN[2] TEXTSET[1]	CANON		U WASH1[2]
DEC VAX/VMS		TEXTSET[1]			LSU
Data General MV					
HP1000					
HP3000					
HP9000 - 200		TEXTSET			
HP9000 - 500		TEXTSET[1]			
IBM MVS					
IBM PC		A-W[1] PERS[1] TEXTSET[1]		PERS	
IBM VM	IAM				
Integrated Solutions					
ICL Perq			GMD		
Prime					
Siemens BS2000					
Sperry 1100					
SUN		TEXTSET[1]			
Texas Instruments PC					

DEC LN03	HP 2680	HP 2688A	HP Laserjet Plus	IBM 38XX, 4250, Sherpa	Imagen	QMS Lasergrafix	Symbolics	Talaris	Xerox Dover	Xerox 2700II	Xerox 9700
					U BC TEXTSET	TEXTSET					U MICH TEXTSET
					OCLC TEXTSET[1]	SCAN LSR TEXTSET					COS SCAN LSR
						U WASH2[1]				BOCHUM	
					STNFD VNDBLT			TALARIS[1]			U DEL
					SRI CLMBIA		U WASH1	TALARIS[1]	CMU	OSU2	
					U MD[2]	TEXTSET U WASH1[2]	U WASH1[2]	TALARIS[1]	STNFD	OSU2	U DEL
K&S PRCYN DEC[2]					K&S[1]	GA TECH T A&M3 TEXTSET	U MASS	TALARIS[1]			ACC TEXTSET
					T A&M1	T A&M1[1]					
	JDJW	JDJW									
	TEXET										
		HP CALTECH									
						T A&M2					
					TEXTSET	GMD		TALARIS[1]			TEXTSET
			TEXTSET[1]		A-W[1] OCLC PERS[1] TEXTSET[1]	A-W[1] PERS[1] TEXTSET[1]					
				SLAC	SLAC	TEXTSET		WASH ST			U DEL TEXTSET
						T A&M3					
						GMD					
					SUN TEXTSET[1]	U DEL TEXTSET					

Figure 2.

Daisy Wheel, Dot Matrix, Electrostatic Printers

[1] *graphics supported*

[2] *included on one of the standard distribution tapes*

	Apple ImageWriter	C Itoh	DEC LP100	Diablo	Epson
Amdahl (MTS)					
Apollo					
Apple Macintosh	K&S[1] A-W[1]				
Cadmus 9200					
CDC Cyber					
DECSYSTEM 10					
DECSYSTEM 20			OSU2		
DEC VAX UNIX					
DEC VAX/VMS		LSU			
Data General MV					
HP1000					JDJW
HP3000				TEXET	USHEF
HP9000 - 200					
HP9000 - 500					
IBM MVS					
IBM PC	MR				A-W, PERS UMILAN1 USHEF
IBM VM					
Integrated Solutions					
ICL Perq					
Prime					
Siemens BS2000					
Sperry 1100					
SUN					
Texas Instruments PC					TA&M1[1]

Facit 4542	Florida Data	Fujitsu	GE 3000	NDK 7700	Printronix	Qume	Texas Instruments 855	Toshiba	Varian	Versatec
			COS							
		U KOLN								
										U KOLN
										GA TECH VNDBLT
	MR								AMS	U WASH1
										U WASH1[2]
INFN[2]								PRCYN	SCI AP	K&S
					T A&M1			T A&M1[1]		T A&M1[1]
						TEXET				
										GMD U MILAN2
								A-W PERS		
				IAM						WZMN
										LLL
					T A&M1[1]		T A&M1[1]			

Figure 3.

Typesetters

[1] *graphics supported*

[2] *included on one of the standard distribution tapes*

[3] *TeX78 support only*

	Allied Linotype CRTronic	Allied Linotype L100, L300P	Allied Linotype L202	Alphatype CRS	Autologic APS-5, Micro-5
Amdahl (MTS)		TEXTSET			TEXTSET
Apollo		TEXTSET			COS SCAN LSR TEXTSET
Apple Macintosh					
Cadmus 9200					
CDC Cyber					
DECSYSTEM 10					
DECSYSTEM 20			ADAPT	AMS	TEXTSET
DEC VAX UNIX		TEXTSET			TEXTSET
DEC VAX/VMS	PRCYN	TEXTSET	PRCYN		INTGRPH[1] TEXTSET
Data General MV					
HP1000					
HP3000					TEXTSET
HP9000 - 200		TEXTSET			
HP9000 - 500		TEXTSET			
IBM MVS					
IBM PC		A-W PERS TEXTSET	PERS		PERS TEXTSET
IBM VM					TEXTSET
Integrated Solutions					
ICL Perq					
Prime					
Siemens BS2000					
Sperry 1100					
SUN		TEXTSET			TEXTSET
Texas Instruments PC					

Compugraphic 8400	Compugraphic 8600	Harris 7500	Hell Digiset
	RECAU[3]		
		SARA	
K&S	K&S		
U SHEF			
	WASH ST		GMD
PERS	PERS		
	WASH ST		
			GMD
	U WISC		

Figure 4.
Video Displays
[1] *graphics supported*
[2] *included on one of the standard distribution tapes*

	AED 483, 512	ANSI-compatible terminals	BBN BitGraph	DEC ReGIS	VAXstation
Amdahl (MTS)					
Apollo					
Apple Macintosh					
Cadmus 9200					
CDC Cyber					
DECSYSTEM 10					
DECSYSTEM 20					
DEC VAX UNIX					
DEC VAX/VMS	ADLD[2]	ADLD[2]		ADLD[2]	TEXTSET
Data General MV					
HP1000					
HP3000					
HP9000 - 200					
HP9000 - 500					
IBM MVS					
IBM PC					
IBM VM					
Integrated Solutions					
ICL Perq					
Prime					
Siemens BS2000					
Sperry 1100					
SUN					
Texas Instruments PC					

DEC VT100	DEC VT125	Tektronix 4014	Visual 500, 550	other screen preview
				YALE TEXTSET
				K&S[1] A-W[1]
				U KOLN
ADLD[2]	INFN[2]	ADLD[2] INFN[2]	ADLD[2]	ADLD[2]
				T A&M1[1]
		U MILAN1		GMD
				A-W PERS TEXTSET
				UCIRV1
				GMD
				GMD
				UCB UCIRV2[2] TEXTSET
				T A&M1[1]

Memory Sizes in Various Implementations of TEX

Barbara Beeton
American Mathematical Society

Probably the most discouraging error message to appear in the middle of a TEX job is "`TeX capacity exceeded, sorry.`" No one is sorrier, of course, than the victim. However, knowing one's requirements and TEX's capacity makes it possible to plan ahead, or at least to be prepared for the eventuality.

The tables on the following pages give the memory capacity compiled into most of the implementations of TEX now in distribution. The values listed are those assigned in §§11–12 of `TeX.WEB`. TEX memory capacity is discussed in some detail on page 300 of *The TEXbook*, and the names given there for segments of memory that can be overloaded (and thus can occur in error messages) are also shown in the tables.

A historical digression is in order. In the original implementation of TEX82, the main dynamic memory was split into two segments, "box memory" (boxes, glue, and paragraph breakpoints) and "macro memory" (token lists and characters). (A diagram of TEX82's memory structure for tables and work areas, but not including the main dynamic memory, appeared in *TUGBOAT* 3, no. 2: 13.) This division proved too limiting — jobs which required a relatively large amount of box memory (such as the output device charts published regularly in *TUGBOAT*) would overload that segment while there was still plenty of untouched macro memory, and vice versa. A major redesign, implemented in version 1.3, combined these two segments; now, capacity is not exceeded until all available "main memory" is exhausted. If you are plagued by `capacity exceeded`, and the message specifies `box memory` or `macro memory`, you are still using a version of TEX older than 1.3, and should (in addition to taking whatever immediate remedies are necessary to complete your current job) do everything you can to get a more up-to-date version installed.

To determine your actual "overhead" requirements, follow the suggestions on page 300 of *The TEXbook*: set `\tracingstats` to a positive value, input the header files you will be using, and say `\bye`; TEX will report the memory required in each category. At some sites, this feature has been turned off to maximize processing speed; if you find this to be true, consult your local wizard.

Examination of the tables will show a wide variation in the memory allotments. The values assigned in the "generic" implementation, `TeX.WEB`, are suitable for general use with `PLAIN.TeX` and can be accommodated by nearly all computer architectures and suitable Pascal compilers. There are many reasons for enlarging the memory allocations; following are some of them.

Large macro packages (e.g. LATEX or AMS-TEX) require significant amounts of memory just to be loaded in, before any user data is input. This has been cited by Kellerman & Smith as the reason for their large string and string pool sizes, and at the AMS is the reason for the pool size increase.

Nearly every implementor seems to agree that increasing the main memory is a good idea. However, some compilers are unable to pack a value greater than 32K into a half-word; TEX permits the main memory starting address to be negative, and the VAX UNIX and DG MV implementations take this option.

At the AMS, some TEX input is program-generated, and input lines can be much longer than those typical of user-keyboarded input.

Bilingual TEX accommodates hyphenation patterns for both English and French, and thus requires additional pattern memory (but not much!). Alternate fonts may also be desirable in a multilingual environment (see the article by Jacques Désarménien, "How to run TEX in a French environment," *TUGBOAT* 5, no. 2: 91–102).

Figure 1.

Memory Sizes.

[1] *Values in brackets are what is used at the AMS, not in the distributed version.*

[2] *Value in brackets = actual used.*

[3] *Values in brackets are for IniTEX.*

[4] *Adjusted at run time for available memory; average sizes in brackets.*

[5] *Dynamically changed depending on system size and user demands; values are typical maximum values using DOS 2.1 on a 640K XT.*

	`TeX.WEB` §11–12	DEC 20 (Stanford, AMS)	VAX UNIX (U of Wash)	VAX/VMS (Stanford)
Version Distributed	2.0	2.0 [1.5][1]	2.0	1.5
Form of Distribution		web, exec	web, exec	web, exec
Compile-Time Parameters				
mem_max main memory	30000	58000	30000	58000
mem_min	0	0	−30000	0
buf_size buffer	500	500 [1500][1]	500	500
error_line	72	79	79	79
half_error_line	42	50	50	50
max_print_line	79	79	79	79
stack_size stack	200	200	200	200
max_in_open text input levels	6	6	15	6
font_max	75	100	100	100
font_mem_size font memory	20000	25000	25000	25000
param_size parameter stack	60	60	60	60
nest_size semantic nest	40	40	40	40
max_strings number of strings	3000	4400	4400	4400
string_vacancies	8000	15000	15000	15000
pool_size pool size	32000	40000 [50000][1]	45000	40000
save_size save size	600	600	600	600
trie_size pattern memory	8000	8000	8000	8000
dvi_buf_size	800	800	800	1024
file_name_size	40	69	1024	69
Parameters Requiring Rerun of `INITEX`				
mem_bot	0	0	−30000	0
mem_top main memory	30000	58000	30000	58000
font_base	0	0	0	0
hash_size hash size	2100	2500	3000	2500
hash_prime	1777	2113	2551	2113
hyph_size exception dictionary	307	307	307	307

VAX/VMS (K&S)	VAX/VMS TeX (INRS)	CDC Cyber (U of Wash)	IBM VM/CMS	Prime	Sperry 1100	Apollo	Data General MV	HP 3000	IBM PC (MicroTeX)	IBM PC (PC TeX)
1.5	1.1	1.5	1.1	1.3	1.3	1.5	1.5	1.5	1.5	1.5
web, exec	web, exec	web	web, exec	web, exec	web, exec	web, exec	web, compiled	exec only	exec only	exec only
65000	65000	32000	60000	58000	58000	60000	30000	65534[4] [58000]	64000[5]	65534[4] [58000]
0	0	0		0	0	0	–30000	0	0	0
500	500	500	1024	500	500	500	500	500	1000	500
79	79	72	79	72	72	79	79	72	79	72
50	50	42	50	42	42	50	51	42	50	42
79	79	79	79	79	79	79	79	79	79	79
200	200	50	200	200	200	200	200	200	200	200
6	6	6	6	6	6	6	6	6	16	6
100	200	75	100	100	75	100	99	75	100	75
25000	50000	18000	25000	25000	20000	25000	25000	30000[4] [26000]	29000[5]	30000[4] [26000]
60	60	60	60	60	60	60	60	60	120	60
40	40	40	40	40	40	40	40	40	40	40
10000	4400	3000	5000	4400	3000	5000	4400	4000	6000[5]	4000
20000	20000	8000	13000	15000	8000[3] [4000]	20000	16000	13000		13000
65000	60000	32000	40000	40000	32000[3] [28000]	50000	40000	40000	60000[5]	40000
600	600	600	600	600	600	600	600	600	600	600
8000	16000[2] [7316]	7000	8000	8000	8000	8000	8000	8000	6000	8000
800	1024	80	2048	800	800	1024	1024	512	1024	512
69	76	26	40	40	88	256	256	40		40
0	n/a	500		0	0	0	–30000	0	0	0
65000	n/a	25000		58000	58000	60000	30000	65534	64000[5]	65534
0	0	0	0	0	0	0	0	0	0	0
5000	2500	2100	2100	2500	2100	3000	2500	2500	2500	2500
4253	2129	1777	1777	2113	1777	2551	2113	2129	2113	2129
307	307	307	307	307	307	307	307	307		307

Sources of Information

`TeX.WEB` Stanford distribution

DEC 20 Stanford distribution

VAX UNIX Pierre MacKay, University of Washington

VAX/VMS David Kellerman, Kellerman & Smith

VAX/VMS Stanford distribution

VAX/VMS TEX Michael Ferguson, INRS

CDC Cyber Jim Fox, University of Washington

IBM VM/CMS Alan Spragens, Stanford Linear Accelerator Center

Prime John Crawford, Ohio State University

Sperry 1100 Bill Kelly, University of Wisconsin, Madison

Apollo Bill Gropp, Yale University

DG MV Bart Childs, Texas A&M University

HP 3000 and IBM PC (PC TEX) Lance Carnes, Personal TEX

IBM PC (MicroTEX) David Fuchs, Stanford University

SMALL TEX IMPLEMENTATIONS

Lance Carnes
Personal TEX

Editor's note: Figure 1 summarizes the available implementations of TEX on small machines. In deciding which implementations to include in the table, a machine was considered "small" if hardware limitations make it difficult to "shoehorn" TEX onto the machine, or if the machine is typically configured for a single user. For instance, the HP-3000 fits because of the first consideration (16-bit word size and small address space), and the Apollo fits because of the second.

The table is mostly the same as appeared in the last *TUGBOAT*. Several implementations that were in progress are now available, but no new entries have been added. The table has been updated to reflect new contacts for several suppliers, and, in some cases, the supplier for a particular implementation has changed.

Those of you working on a version of TEX for a small machine (you with Amigas and Ataris, for instance) should contact Lance Carnes or Barbara Beeton so they can include your implementation in the next version of the table.

Figure 1.
TEX Systems for Small Computers.

Apollo MC68000	Gilbert Gingras 514-738-2191	COS Information 5647, rue Ferrier Montréal, Québec H4P 1N1
	Thom Hickey 614-764-6075	OCLC Box 7777 Dublin, OH 43017
	Bruce Baker 313-996-3566	Textset Box 7993 Ann Arbor, MI 48107
	Bill Gropp 203-436-3761	Yale University Department of Computer Science Box 2158, Yale Station New Haven, CT 06520
Apple Macintosh MC68000	Brenda Cavallaro 617-844-6795	Addison-Wesley Publishing Reading, MA 01867
Cromemco CS300 MC68000	Brad Finney 707-826-3619	Humboldt State University Department of Engineering Arcata, CA 95521
Cyb MC68000	Norman Naugle 409-845-3104	Texas A&M University Mathematics Department College Station, TX 77843
HP 1000	John Johnson 415-965-3245	JDJ Wordware Box 354, Cupertino, CA 95015
HP 3000	Lance Carnes 415-388-8853	TeXeT 163 Linden Lane Mill Valley, CA 94941
IBM PC, XT, AT 8088, 80286	Lance Carnes 415-388-8853	Personal TEX 20 Sunnyside, Suite H Mill Valley, CA 94941
	Toni Formichella 617-944-3700x2640	Addison-Wesley Publishing Educational Media Systems Reading, MA 01867
IBM XT, AT 8088, 80286	Ronny Bar-Gadda 415-326-1275	446 College Avenue Palo Alto, CA 94306
Masscomp MC68000	Norman Naugle 409-845-3104	Texas A&M University Mathematics Department College Station, TX 77843
PERQ/PNX	Wolfgang Appelt	Gesellschaft für Mathematik und Datenverarbeitung D-5202 Sankt Augustin Federal Republic of Germany
Sun Microsystems MC68000	Bruce Baker 313-996-3566	Textset Box 7993 Ann Arbor, MI 48107
	Pierre MacKay 206-545-2386	University of Washington Computer Science, FR-35 Seattle, WA 98195
Synapse MC68000	Dick Wallenstein 609-764-1720	Comcon 5 Underwood Court Delran, NJ 08075

Institutional Members

Aarhus Universitet, Det Regionale EDB-Center (RECAU), *Aarhus, Denmark*

Addison-Wesley Publishing Company, *Reading, Massachusetts*

American Mathematical Society, *Providence, Rhode Island*

ASCII Corporation, *Tokyo, Japan*

Baumeister College, *Witten, West Germany*

Bell Northern Research, Inc., *Mountain View, California*

California Institute of Technology, Computer Science Library, *Pasadena, California*

CALMA, *Sunnyvale, California*

Calvin College, *Grand Rapids, Michigan*

Canon, Inc., Office System Center, *Tokyo, Japan*

Carleton University, *Ottawa, Ontario, Canada*

CDS/WordWorks, *Davenport, Iowa*

Centre Inter-Régional de Calcul Électronique, CNRS, *Orsay, France*

College of St. Thomas, *St. Paul, Minnesota*

Columbia University, Center for Computing Activities, *New York City*

Corbel & Co., *Jacksonville, Florida*

COS Information, *Montréal, Québec, Canada*

Digital Equipment Corporation, *Nashua, New Hampshire*

Educational Testing Service, *Princeton, New Jersey*

Electricité de France, *Clamart, France*

European Southern Observatory, *Garching bei München, West Germany*

Geophysical Company of Norway A/S, *Stavanger, Norway*

Georgia Institute of Technology, School of Information & Computer Science, *Atlanta, Georgia*

Grumman Corporation, *Bethpage, New York*

GTE Laboratories, *Waltham, Massachusetts*

Harvard University, Computer Services, *Cambridge, Massachusetts*

Hewlett-Packard Co., *Boise, Idaho*

IBM Corporation, Scientific Center, *Palo Alto, California*

Illinois Institute of Technology, Academic Computing Center, *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*

Istituto di Cibernetica, Università degli Studi, *Milan, Italy*

Los Alamos National Laboratory, University of California, *Los Alamos, New Mexico*

Marquette University, Department of Mathematics, Statistics and Computer Science, *Milwaukee, Wisconsin*

Massachusetts Institute of Technology, Artificial Intelligence Laboratory, *Cambridge, Massachusetts*

Mathematical Reviews, American Mathematical Society, *Ann Arbor, Michigan*

McGill University, Computing Centre, *Montreal, Quebec, Canada*

McGraw-Hill, Inc., *Englewood, Colorado*

Microelectronics Center of North Carolina, *Research Triangle Park, North Carolina*

National Center for Atmospheric Research, *Boulder, Colorado*

National Institutes of Health, *Bethesda, Maryland*

National Research Council Canada, *Ottawa, Ontario, Canada*

Online Computer Library Center, Inc. (OCLC), *Dublin, Ohio*

QMS, Inc, *Mobile, Alabama*

Queens College, *Flushing, New York*

Reed College, *Portland, Oregon*

RE/SPEC, Inc., *Rapid City, South Dakota*

Rubicon Group, Ltd, *Austin, Texas*

Ruhr Universität Bochum, Rechenzentrum, *Bochum, West Germany*

Sandia National Laboratories, *Albuquerque, New Mexico*

SAS Institute, *Cary, North Carolina*

Schlumberger Offshore Services, *New Orleans, Louisiana*

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*

Institutional Members

Software Research Associates,
Tokyo, Japan

Sony Corporation, *Atsugi, Japan*

Springer-Verlag,
Heidelberg, West Germany

Stanford Linear Accelerator Center (SLAC), *Stanford, California*

Stanford University,
ITS Graphics & Computer Systems,
Stanford, California

State University of New York,
Stony Brook, New York

Syracuse University,
Syracuse, New York

Talaris Systems, Inc.,
San Diego, California

Texas A&M University,
Department of Computer Science,
College Station, Texas

Texas A&M University,
Department of Mathematics,
College Station, Texas

Textset, Inc., *Ann Arbor, Michigan*

TRW, Inc.,
Redondo Beach, California

Tufts University,
Mathematics Department,
Medford, Massachusetts

TYX Corporation, *Reston, Virginia*

University of British Columbia,
Vancouver, British Columbia, Canada

University of Calgary,
Academic Computing Services,
Calgary, Alberta, Canada

University of California, Berkeley,
Computer Science Division,
Berkeley, California

University of California, Berkeley,
Computing Services,
Berkeley, 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 Delaware,
Academic Computing Services,
Newark, Delaware

University of Glasgow,
Dept of Computing Science,
Glasgow, Scotland

University of Groningen,
Groningen, The Netherlands

University of Maryland,
College Park, Maryland

University of Massachusetts,
Department of Computer and Information Science,
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 Texas at Austin,
Physics Department, *Austin, Texas*

University of Texas at Dallas,
Center for Space Science,
Dallas, Texas

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

University of York, *York, England*

Vanderbilt University,
Computer Center,
Nashville, Tennessee

Washington State University,
Computing Service Center,
Pullman, Washington

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
	1986 TUGboat Subscription/TUG Membership (Jan.–Dec.) – **North America** New (first-time): [ ] $25.00 each Renewal: [ ] $35.00; [ ] $25.00 – reduced rate if renewed before January 31, 1986	
	1986 TUGboat Subscription/TUG Membership (Jan.–Dec.) – **Outside North America** New (first-time): [ ] $30.00 each Renewal: [ ] $40.00; [ ] $30.00 – reduced rate if renewed before January 31, 1986	
	TUGboat back issues, $15.00 per issue, circle issue(s) desired: 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	

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.

TEX82 Order Form

The latest official versions of TEX software and documents are available from Maria Code by special arrangement with the Computer Science Department of Stanford University.

Ten different tapes are available. The generic distribution tape contains the source of TEX82, WEB, and the latest (prototype) version of WEB **METAFONT**, standard test programs for TEX and **METAFONT**, a few "change" files, the collection of fonts in TFM format, and other miscellaneous materials; a PASCAL compiler will be required to install programs from a generic tape. The TEX distribution tapes include the AMS-TEX, LATEX and HP TEX macro packages; other macro packages will be added as they become available. The special distribution tapes are for the indicated systems only, and should be ordered for these systems instead of a generic tape. Two tapes are PXL font collections covering various magnifications at 200/240 dots/inch and 300 dots/inch respectively.

Each tape will be a separate 1200 foot reel which you may send in advance or purchase (for the tape media) at $10.00 each. Should you send a tape, you will receive back a different tape. Tapes may be ordered in ASCII or EBCDIC characters. All tapes are 1600 bpi.

The tape price of $82.00 for the first tape and $62.00 for each additional tape (ordered at the same time) covers the cost of duplication, order processing, domestic postage and some of the costs at Stanford University. Extra postage is required for first class or export.

Manuals are available at the approximate cost of duplication and mailing. Prices for manuals are subject to change as revisions and additions are made.

Please send a money order or check (drawn on a US bank) along with your order if possible. Your purchase order will be accepted, as long as you are able to make payment within 30 days of shipment. Make checks payable to Maria Code.

The order form contains a place to record the name and address of the person who will actually use the TEX tapes. This should *not* be someone in the purchasing department.

Your order will be filled with the most recent versions of software and manuals available from Stanford at the time your order is received. If you are waiting for a specific release, please indicate this. Orders are normally filled within a few days. There may be periods (like short vacations) when it will take longer. You will be notified of any serious delays. If you want to inquire about your order you may call Maria Code at (408) 735-8006 between 9:30 a.m. and 2:30 p.m. West Coast time.

If you have questions regarding the implementation of TEX or the like, you must take these to Stanford University or some other friendly TEX user.

Now, please complete the order form on the reverse side.

1/86

TEX82 Order Form

TAPES:

TEX generic distribution tapes (PASCAL compiler required):

________	† ASCII format	________	EBCDIC format

TEX distribution tapes in special formats:

________	† VAX/VMS Backup format	________	IBM VM/CMS format
________	† DEC 20/Tops-20 Dumper format	________	* IBM MVS format

† Includes WEB **METAFONT** * Not yet available; call before ordering

Font tapes:

________	Font library (200/240 dots/inch)	________	Font library for IBM 4250 printer
________	Font library (300 dots/inch)	________	Font library for IBM 6670 printer

________ Total number of tapes.

Tape costs: $82.00 for first tape; $62.00 for each additional.

Tape cost = $ ________

Media costs: $10.00 for each tape required. Media cost = $ ________

MANUALS:

________	TEX: The Program – $28.00	________	A Torture Test for TEX – $8.00
________	WEB – $10.00	________	TEXware – $8.00
________	TEXbook – $20.00	________	LATEX – $20.00
________	BibTEX– $10.00		

Manuals cost = $ ________

California orders only: add sales tax = $ ________

Domestic book rate: no charge.
Domestic first class: $2.50 for each tape and each manual.
Export surface mail: $2.50 for each tape and each manual.
Export air mail to Canada & Mexico: $4.00 each.
Export air mail to Europe: $7.00 each.
Export air mail to other areas: $10.00 each. Postage cost = $ ________

(Make checks payable to Maria Code) Total order = $ ________

Name and address for shipment: Person to contact (if different):

________________________________ ________________________________
________________________________ ________________________________
________________________________ ________________________________
________________________________ ________________________________

Telephone ________________

Send to: Maria Code, DP Services, 1371 Sydney Dr., Sunnyvale, CA 94087

1/86

TEX-Related Publications Available from TUG

For prices and ordering instructions, see separate order form. Current forms may be obtained by writing to TUG, P. O. Box 9506, Providence, RI 02940, U.S.A., or calling (401) 272-9500, ext. 232.

Computers and Typesetting by Donald E. Knuth
A hardcover library of five volumes containing the authoritative documentation on TEX and METAFONT. Volumes A–E are described below.

Published by Addison-Wesley Publishing Co., Inc., Reading, Mass., 1986.

The TEXbook by Donald E. Knuth
From the cover: "TEX represents the state-of-the-art in computer typesetting. It is particularly valuable where the document, article, or book to be produced contains a lot of mathematics, and where the user is concerned about typographic quality. TEX software offers both writers and publishers the opportunity to produce technical text, in an attractive form, with the speed and efficiency of a computer system.

"Novice and expert users alike will gain from *The TEXbook* the level of information they seek. ... The novice need not learn much about TEX to prepare a simple manuscript with it. But for the preparation of more complex documents, *The TEXbook* contains all the detail required."

Volume A of **Computers and Typesetting**. Published jointly by Addison-Wesley Publishing Co., Inc., Reading, Mass., and American Mathematical Society, Providence, R.I., 1986 edition. (Also available in softcover.)

TEX: The Program by Donald E. Knuth
From the introduction: "The main purpose of the... program is to explain the algorithms of TEX as clearly as possible. As a result, the program will not necessarily be very efficient when a particular Pascal compiler has translated it into a particular machine language. However, the program has been written so that it can be tuned to run efficiently in a wide variety of operating environments by making comparatively few changes. Such flexibility is possible because the documentation... is written in the WEB language, which is at a higher level than Pascal; the preprocessing step that converts WEB to Pascal is able to introduce most of the necessary refinements. Semi-automatic translation to other languages is also feasible, because the program... does not make extensive use of features that are peculiar to Pascal."

Volume B of **Computers and Typesetting**. Published by Addison-Wesley Publishing Co., Inc., Reading, Mass., 1986.

The METAFONTbook by Donald E. Knuth
From the preface: "METAFONT is a system for the design of alphabets suited to raster-based devices that print or display text. The characters [used to set the book] were all designed with METAFONT, in a completely precise way; and they were developed rather hastily by the author of the system, who is a rank amateur at such things. It seems clear that further work with METAFONT has the potential of producing typefaces of real beauty. This manual has been written for people who would like to help advance the art of mathematical type design."

Volume C of **Computers and Typesetting**. Published jointly by Addison-Wesley Publishing Co., Inc., Reading, Mass., and American Mathematical Society, Providence, R.I., 1986; available Summer '86. (Also available in softcover.)

METAFONT: The Program by Donald E. Knuth
The complete source code for METAFONT.

Volume D of **Computers and Typesetting**. Published by Addison-Wesley Publishing Co., Inc., Reading, Mass., 1986; available Summer '86.

Computer Modern Typefaces by Donald E. Knuth
This reference work contains the full METAFONT descriptions of the letters and symbols which comprise the Computer Modern family of typefaces. The preface to an earlier report on this subject stated, "It is hoped that some readers will be inspired to make similar definitions of other important families of fonts. The bulk of this [document] consists of... METAFONT programs for the various symbols needed, and as such it is pretty boring, but there are some nice illustrations."

Volume E of **Computers and Typesetting**. Published by Addison-Wesley Publishing Co., Inc., Reading, Mass., 1986; available Summer '86.

TUGBOAT

TUGboat is the newsletter of the TEX Users Group. It contains communications from the Stanford TEX Project; articles of interest to installers and users of TEX and METAFONT; reports of activity at distribution centers and user sites; macros, problems, questions and answers; a calendar of TEX and TUG-related events; and official TUG business.

Three issues are planned for 1986. Memberships and subscriptions are accepted on a calendar-year basis only. All back issues are available.

First Grade TEX: A Beginner's TEX Manual
by Arthur L. Samuel
From the introduction: "This is an introductory ready-reference TEX82 manual for the beginner who would like to do First Grade TEX work. Only the most basic features of the TEX system are discussed in detail. Other features are summarized in an appendix and references are given to the more complete documentation available elsewhere."

Originally published as Stanford Computer Science Department Report No. STAN-CS-83-985. Reprinted with permission and distributed by TUG.

The Joy of TEX by Michael Spivak
This is the user's guide for AMS-TEX, an extension of Donald Knuth's typesetting program TEX. AMS-TEX and *The Joy of TEX* were written to allow simplified input of mathematical material, and to permit the output to be formatted according to various preset style specifications. Use of AMS-TEX requires no expertise in mathematics or in computer science; it requires no more than that TEX itself be available.

Published by the American Mathematical Society, Providence, R.I., 1986.

LATEX: A Document Preparation System
by Leslie Lamport
From the preface: "The LATEX document preparation system is a special version of Donald Knuth's TEX program. ... LATEX adds to TEX a collection of commands that simplify typesetting by letting the user concentrate on the structure of the text rather than on formatting commands. In turning TEX into LATEX, I have tried to convert a highly-tuned racing car into a comfortable family sedan. The family sedan isn't meant to go as fast as a racing car or be as exciting to drive, but it's comfortable and gets you to the grocery store with no fuss. However, the LATEX sedan has all the power of TEX hidden under its hood, and the more adventurous driver can do everything with it that he can with TEX."

Published by Addison-Wesley Publishing Co., Inc., Reading, Mass., 1985.

An Introduction to LATEX by Michael Urban
From the introduction: "This paper is intended to introduce you to the LATEX document preparation system, and to get you working with LATEX as fast as possible. It is *not* a complete reference work on LATEX."

Originally prepared for the TRW Software Productivity Project, 1986. Reprinted with permission and distributed by TUG.

LATEX Command Summary by L. Botway and C. Biemesderfer
Introduction: "This summary ostensibly represents LATEX control sequences for the software implemented at STScI [Space Telescope Science Institute] as of December 6, 1985 (TEX 1.3, LATEX 2.08)."

Originally prepared for internal use at STScI; reprinted with permission and distributed by TUG.

VAX Language-Sensitive Editor (LSEDIT) Quick Reference Guide for use with the LATEX Environment
From the statement of scope and intent: "This guide presents information about the Language-Sensitive Editor (LSEDIT) and the environment defined for LATEX. It is a supplement to the *VAX Language-Sensitive Editor User's Guide* and *LATEX, A Document Preparation System, User's Guide & Reference Manual.* This guide acts as a summary and memory refresher for the commands and functions covered in [these manuals]. It is *not* intended to replace either of [them]."

Originally prepared for internal use at Lear Siegler, Inc., Instrument Division, Grand Rapids, Michigan; reprinted by permission and distributed by TUG.

Proceedings of the First European Conference on TEX for Scientific Documentation, Como, Italy, May 16–17, 1985 Dario Lucarella, Editor
From the preface: "The aim of the Conference was to provide a state-of-the-art survey of current research activities and the latest applications that are growing around TEX. The topics covered ... concern the following fields: Documentation systems based on TEX; TEX as a tool for authors; Customization of TEX for non-English languages; Hyphenation; Standardization problems; Facilities for interactive entering and retrieving of formulae; METAFONT and font design; Implementation of TEX, METAFONT and drivers." All papers are in English.

Published by Addison-Wesley Publishing Co., Inc., Reading, Mass., 1985.

Mathematics into Type by Ellen Swanson
This book covers the publication of mathematics from manuscript to printed book or journal, with emphasis on the preparation of copy for the compositor and the proofreading and makeup of the publication. It will be useful to the author who is directly concerned in the editing of his book, and it should benefit any author who is preparing a manuscript for publication.

Published by the American Mathematical Society, Providence, R.I., 1982.

Personal TEX Inc

Your **Complete** Supplier of TEX for the PC!

... now offers a list of software, fonts and hardware so that we can be your complete TEX outfitter for PC and AT workstations. We have joined forces with Textset n^2 Computer Consultants, and the Metafoundry to bring you these products:

SOFTWARE:

PCTEX A full TEX82, version 1.5, including INITEX, LATEX 2.09, LATEX User's Guide AMS-TEX, and Mike Spivak's PCTEX Manual and VANILLA macro package. **$249.**

PCDOT Device drivers for the following dot-matrix printers: Epson FX, RX and LQ printers, IBM Graphics Printer, and the Toshiba 1340, 1350, P351 printers. Each driver includes over 230 TEX and LaTEX fonts. **$100. each.**

PCLaser Device drivers for several popular laser printers: Apple **LaserWriter (Postscript)** (Textset), QMS **Lasergrafix 800, 1200** (n^2 Computer Consultants, Textse **Imagen 8,12,24/300** (Textset), **Corona LP300** and **JLASER (HP LaserJet)**. Each driver includes a complete set of TEX and LaTEX fonts. **$300. each.**

Preview Textset's popular Preview is now available for the PC. Requires the Hercule Graphics Card, IBM Enhanced Graphics Adapter, or Tecmar Graphics Master. **$250.**

FONTS:

MF Medley 44 Popular fonts from the Metafoundry library. Includes printer and scre preview resolutions of Chel fonts (Computer Helvetica, shown here), and Copperplate, Blacl Letter and Schoolbook headline fonts. Specify printer resolution (180, 240 or 300 DPI) **$10**

TEX Filters Interfaces from PC Paintbrush, PC Palette, FancyFont and FONTRIX TEX-compatible font files. An easy way to incorporate graphics with TEX. **$100.**

HARDWARE:

Corona Laser Printer This device will print a full page of TEX output, and employs the same Canon CX print engine found in popular laser printers. Comes with PCTEX and the Corona device driver. Complete software and hardware: only **$2995.**

JLASER An interface card for your PC with 1 MB RAM and a direct connection to Canon CX print engine (HP LaserJet, QMS KISS and others). Allows high resolution full page printing. From **$699.**

Join hundreds of satisfied PCTEX users. Write or call us today for complete product informati Inquire about educational and corporate discounts and site licensing.

System requirements: DOS 2.0 or better, 512K RAM, 10M hard disk. Preview requires appropriate graphics apadter. Corona La Printer requires additional 512K RAM disk. Include $8. shipping and handling for each U.S. order. (Shipping to Canada: $15. International Air Mail: $40.) California orders, add 6% sales tax. MasterCard, Visa accepted.

Personal TEX Inc

20 Sunnyside, Suite H
Mill Valley, CA 94941
(415) 388-8853 Telex 510-601-0672

Distributors: **Canada:** Ahearn & Soper, Rexdale, Ontario, (416) 675-3999; **UK:** UniTEX Systems, Sheffield, (0742) 351 489; **Germany:** Kettler EDV-Consulting, Lenggries, (08042) 8081; **France:** Probe Informatique, Trappes, (062) 2603; **Ireland:** UniTEX, Dublin, 772941 x1983;

This ad was generated using PCTEX, and printed on a Corona Laser Printer.

The Joy of TeX

A Gourmet Guide to Typesetting with the $\mathcal{AMS}$-TeX macro package

M. D. SPIVAK, Ph.D.

The Joy of TeX is the user-friendly user's guide for $\mathcal{AMS}$-TeX, an extension of TeX, Donald Knuth's revolutionary program for typesetting technical material. $\mathcal{AMS}$-TeX was designed to simplify the input of mathematical material in particular, and to format the output according to any of various preset style specifications.

There are two primary features of the TeX system: it is a computer system for typesetting technical text, especially text containing a great deal of mathematics; and it is a system for producing beautiful text, comparable to the work of the finest printers.

Most importantly, TeX's capabilities are not available only to TeXperts. While mathematicians and experienced technical typists will find that TeX allows them to specify mathematical formulas with greater accuracy and still have great control over the finished product, even novice technical typists will find the manual easy to use in helping them produce beautiful technical TeXt.

This book is designed as a user's guide to the $\mathcal{AMS}$-TeX macro package and details many features of this extremely useful text processing package. Parts 1 and 2, entitled "Starters" and "Main Courses," teach the reader how to typeset most normally encountered text and mathematics. "Sauces and Pickles," the third section, treats more exotic problems and includes a 60-page dictionary of special TeXniques.

Exercises sprinkled generously through each chapter encourage the reader to sit down at a terminal and learn through experimentation. Appendixes list summaries of frequently used and more esoteric symbols as well as answers to the exercises.

AMERICAN MATHEMATICAL SOCIETY · FOUNDED 1888 · ΑΓΕΩΜΕΤΡΗΤΟΣ ΜΗ ΕΙΣΙΤΩ

ISBN 0-8218-2999-8, LC 85-7506
290 pages, Due April 1986
AMS Indiv. Memb. $24, AMS Inst. Memb. $28, List price $32
To order specify JOYT/T

Shipping/Handling: 1st book $2, each add'l $1, max. $25; by air, 1st book $5, each add'l $3, max. $100

PREPAYMENT REQUIRED. Order from
American Mathematical Society
PO Box 1571
Annex Station
Providence, RI 02901-1571
or call 800-556-7774 to use VISA or MasterCard.

C O M P L E T E
TYPESETTING
S E R V I C E S

Math and Technical Textbook Publishers . . .

If you are creating your book files with Donald Knuth's TEX software package, Computer Composition Corporation can now offer the following specialized services:

— Converting TEX DVI files to the fully paginated typeset page. TEX input files can be submitted to us on diskettes or magnetic tape.

— Providing laser-printed page proofs which very closely simulate the final typeset page.

— Keyboarding from traditionally-prepared manuscripts via the TEX processing system.

— Camera work, including the shooting of halftones, screens, line art, etc.

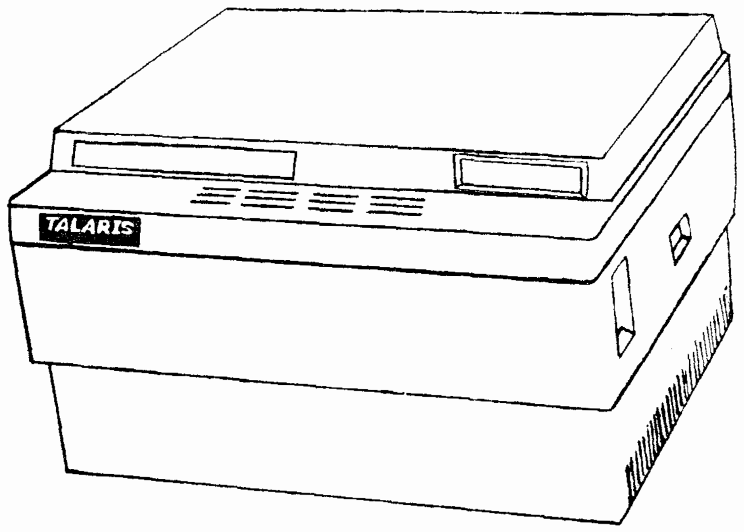
Call or write us for sample pages and proofs.

COMPUTER COMPOSITION CORPORATION
1401 WEST GIRARD •• MADISON HEIGHTS, MICHIGAN 48071 •• (313) 545-4330

TALARIS SYSTEMS INC.®

THE TALARIS 810:

THE MISSING PIECE IN YOUR PERSONAL TYPESETTING SYSTEM



Now that you have TeX™ on your IBM PC or compatible, you need a printer that lives up to TeX's formatting standards—a high-resolution printer with fonts and flexibility. What you don't need is a printer that costs far more than your PC.

Introducing the Talaris 810

The *Talaris 810* is that printer—an 8-page-per-minute, 300 dots-per-inch laser printer that costs just $3,850. Ideal for TeXnicians on a budget.

And Talaris completes your personal typesetting system with fonts and software: TeX PROM font cartridges and a DVI file post-processor.

Talaris's TeX font cartridges

The *Talaris 810* comes with 12 standard word processing fonts. Add one of Talaris's TeX font cartridges and start TeXing. Each font cartridge contains from 16 to 30 TeX fonts. Choose from Plain, Plain+ (30 fonts), Dutch Times Roman-like, or Swiss Helvetica-like TeX font cartridges.

Talaris's LoTeX DVI post-processor

LoTeX outputs ANSI 3.64 commands to support absolute character positioning as required by TeX DVI files. LoTeX also provides three options for positioning the page on the paper—centered, uncentered, and TeX standard—as well as options for landscape orientation and printing more than one copy of output.

The Talaris 810 and graphics

Store digitized artwork, forms, and signatures as overlays. Then use TeX's \special command to call the overlays.

Talaris, the personal typesetting software specialists

Talaris has the software to keep your *Talaris 810* printing up to speed. Talaris also offers the *Talaris 810* and LoTeX for VAX/VMS systems. And, for vector graphics and higher volume printing, Talaris offers a family of full-page bit-map laser printers and software.

For more information, call or write Talaris Systems Inc., 5160 Carroll Canyon Road, P.O. Box 261580, San Diego, CA 92126; (619) 587-0787.

This ad was created with TeX and Talaris's Dutch fonts; the original was printed on a Talaris 810 laser printer.

LETTERSPACING is available for TEX users!

In the sophisticated printing environment of today, it is a MUST to produce output with letterspacing. Until recently, letterspacing was not available in the existing drivers. Word spaces were sometimes too large in the DVI file from TEX because of hyphenation problems caused by foreign words, very long words or very narrow margins. We now announce that letterspacing is available for TEX users who want to produce high quality output using TEX.

Here is an example of text without letterspacing:

The most famous balneological health resorts abroad are Bad Elster, Brambach, and Wiesenbad in the German Democratic Republic (GDR); Ciechocinek in Poland; Borsec and Băile Herkulane in Rumania; Karlovy Vary, Mariánské Lázně, Piešt'any, Poděbrady, and Františkovy Lázně in Czechoslovakia; Vrnjačka Banja in Yugoslavia.

The chief natural therapeutic factor of a health resort determines the resort's basic classification: balneological, peloid, or climatic. If a resort offers several therapeutic factors, it is classified as climatobalneological, balneopeloid, climatopeloid, or climato-balneopeloid.

This is the same text using letterspacing:

The most famous balneological health resorts abroad are Bad Elster, Brambach, and Wiesenbad in the German Democratic Republic (GDR); Ciechocinek in Poland; Borsec and Băile Herkulane in Rumania; Karlovy Vary, Mariánské Lázně, Piešt'any, Poděbrady, and Františkovy Lázně in Czechoslovakia; Vrnjačka Banja in Yugoslavia.

The chief natural therapeutic factor of a health resort determines the resort's basic classification: balneological, peloid, or climatic. If a resort offers several therapeutic factors, it is classified as climatobalneological, balneopeloid, climatopeloid, or climato-balneopeloid.

Notice that the ligatures were output as separate characters when the lines were letterspaced.

We currently use the letterspace option in our Autologic APS µ5 driver which uses the International 3 (A7) layout with floating accents. We are presently developing drivers for other typesetters and laser printers incorporating this extra polish.

For more information please contact:

Martin Boros
Software engineer

COS
COS INFORMATION
5647, rue Ferrier
Montréal, Québec
H4P 1N1
(514) 738-2191

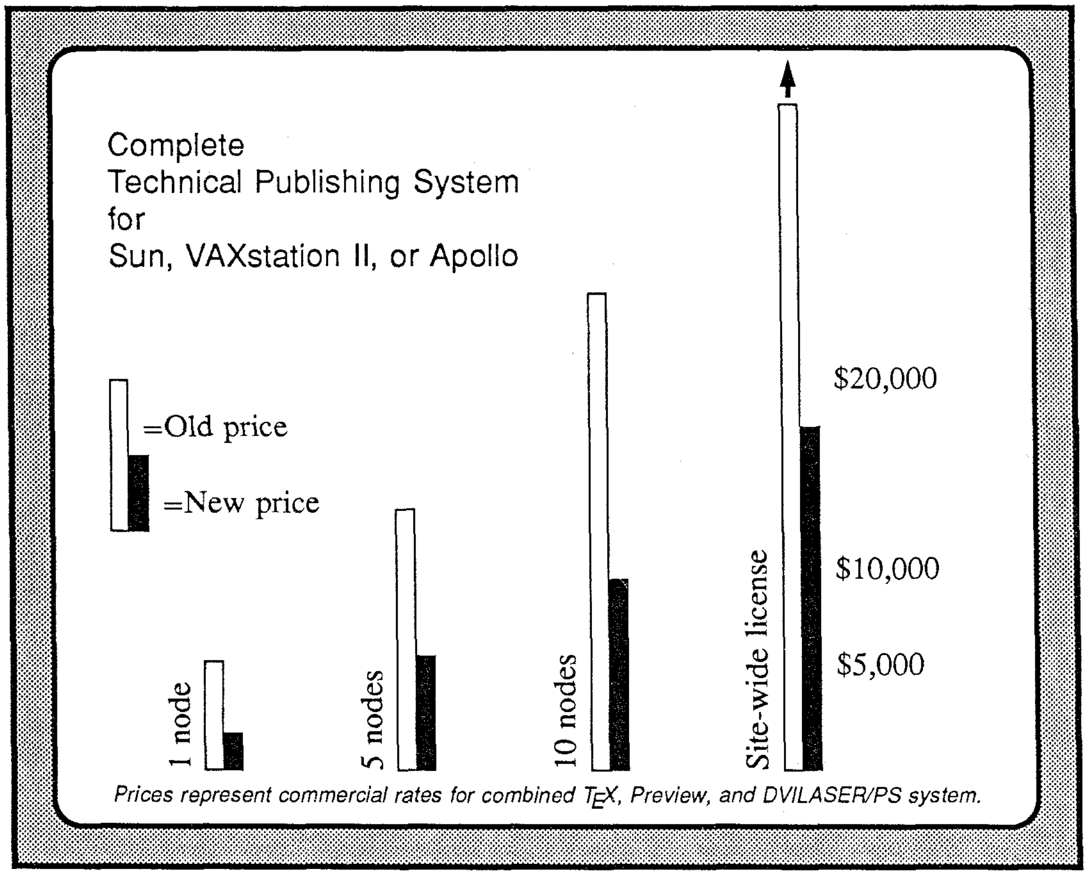
TEX µ5 driver:

– PC	$395.00
– Apollo	$1,195.00
– MicroVax	$1,195.00

TEXTSET, Inc. 416 Fourth Street • P.O. Box 7993 • Ann Arbor • MI • 48107 • (313) 996-3566

technical composition software and services

Announcing New Low Prices



Prices represent commercial rates for combined TEX, Preview, and DVILASER/PS system.

A full TEX system including TEX, Preview, and DVILASER/PS for PostScript printers now costs less than fifty percent of last year's price. Educational and large site discounts are even more dramatic.

We invite you to compare our technical publishing software with all of the available alternatives. For professional, book-quality documents combining mathematics, graphics, tables, and complicated page layouts you won't find a better known and more widely used solution than TEX.

TEXTSET's products now run on systems ranging from 8086 family MSDOS microcomputers, to desktop engineering workstations, to IBM mainframes. We support many popular laser printers and phototypesetters. Whether your system is large or small, TEX produces the same, high quality, printed pages.

This flyer was composed with TEX and PostScript on a Sun II Workstation and printed on an Apple LaserWriter.

ANNOUNCING...

MACROTEX A TEX toolkit. An extensive set of macros that function in a Plain TEX environment. The functionality of a full macro package without restricting your complete access to TEX, compatible with preexisting macros or macros yet to be designed.

Table of Contents Generation ◇ *Index production* ◇ *Cross Referencing* ◇ *Symbolic names for Equations* ◇ *Bibliography production* ◇ *Paper, Letter Formats* ◇ *Autonumbering Chapters, Sections, Equations, Figures* ◇ *Figure macros* ◇ *Margin notes, boxed and plain* ◇ *Inserts: double column, partial page* ◇ *Itemization, automatically numbered or lettered* ◇ *Table Macros* ◇ *Diagonal lines* ◇ *Diamond shapes of any size or slant* ◇ *And many more...*

Documentation includes directions for MACROTEX use as well as the complete MACROTEX code annotated with suggestions for customization. Diskette included for PC users. Projected date of release: May 1, 1986. Price: $50 · Please write or call for complete descriptive brochure and order form.

TEXnology, Inc. *Amy Hendrickson, Consultant*

57 Longwood Avenue, Brookline, MA 02146 *(617) 738-8029*

Custom Macro Writing · *Instruction* · *Book Production*

Index of Advertisers