Why TEX Should NOT Output PostScript — Yet: Addendum

Shane Dunne
University of Western Ontario

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

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

Additional Reference:

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

ASCII Preview with vuTEX

Warren Wolfe

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

Features

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

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

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

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

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

Overview of vuTEX

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

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

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

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

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

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

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

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

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

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

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

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

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

If the compress mode is selected, an effort is made to eliminate blank space so that the output is narrower and a sense of the proper ratio of black type to white space is presented. First, all associated sub/superscripts are squeezed into available space in the base line. Although this may be modified later to preserve the off-the-line printing, vuTEX's current algorithm is unable to drag the associated sub/superscript along with the characters in the base line as shifts are made to close the gaps. Thus, lines are compressed vertically first so that

off-the-line script may appear in the base line beside the associated character or word. During horizontal compression, the first word in each line retains its original spacing in the line so that left indentation is maintained. All other characters are shifted left so that a single space appears between words. Of course, this destroys most vertical alignment, and deliberate horizontal spacing disappears. However, the effect is often a more readable version, particularly with ordinary text. See Example 2.

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

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

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

Conclusion

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

```
 Example       1.

*         EXERCISE8.11

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


          Page    168  of  the  T  Xbook.
-                                 E
```

```
 Example 2.

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

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

 Page 1 of the TE Xbook.
```