

* * * * *

Warnings & Limitations

* * * * *

Another Hangup

In the last issue, you were warned that repetition of a `\let` statement can cause `TEX` to hang. It has been pointed out that `\ifx` can be used to detect recursion (provided you are using a recent enough version of `TEX`; see the errata list, extensions since June 30, 1981).

Another way to make `TEX` hang is to set to a negative value one of `TEX`'s integer parameters that is expecting a positive value. For example, `\chpar3` and `\chpar13` (which will become `\penpen`, `\dhp` and `\adjpen` in `TEX82`) have caused the SAIL version of `TEX` to loop at the Math Society (we are still running with a version of March 1981). This will probably be trapped in `TEX82`, but I didn't see it described in the differences list.

Barbara Beeton

* * * * *

MACRO
O
L
U
M
N

Send Submissions to:
Lynne A. Price
TUG Macro Coordinator
Calma R&D
212 Gibraltar Dr.
Sunnyvale, CA 94086

In the last issue of TUGboat (Volume 2, No. 3) Michael Plass described macros for producing syntax diagrams. The package includes macros for automatically allocating box and counter numbers, much as Patrick Milligan's `\DefineFont` macro (TUGboat, Volume 2, No. 2) assigns font codes. Mike's allocation macros are

```
\def\Alloc#1#2{\def#2{#1}}
\def\AllocBox
{\def\AllocBox{\def\AllocBox{\def\AllocBox
{\def\AllocBox{\def\AllocBox{\def\AllocBox
{\def\AllocBox{\def\AllocBox{\def\AllocBox
{\Overflow
}\Alloc9}\Alloc8}\Alloc7}\Alloc6}\Alloc5
}\Alloc4}\Alloc3}\Alloc2}\Alloc1}
\def\AllocCtr
{\def\AllocCtr{\def\AllocCtr
{\def\AllocCtr{\def\AllocCtr
{\Overflow}\Alloc8}\Alloc7}\Alloc6}\Alloc5}
```

An example of using these macros to select a counter number is

```
\AllocCtr\counternumber
\setcount\counternumber 0
```

The first time `\AllocCtr` is called, it executes the second `\def\AllocCtr` (thus redefining itself to be the text beginning with the third `\def\AllocCtr` and ending with `Alloc 6`) and calls `Alloc` with a first parameter of 5. The second time it is called, it redefines itself removing another nested definition and passing the value 6 to `Alloc`. Similarly, `\AllocBox` successively returns the digits 1 through 9.

* * * * *

TUGBOAT MACRO INDEX

The following list catalogues macros that have appeared in TUGboat. Entries are listed by volume, number, and page as well as author's name. Items that could not be categorized by an obvious headword have been listed under "miscellaneous". Many items refer to parts of large macro packages; users of other packages may find them valuable models for macros of their own.

Readers' comments on the format as well as the contents of this index are welcome.

ACM style	II:1 61, 82-83	A. Keller
Addresses	II:1 54	B. Beeton
.	II:2 A-35	M. Diaz
Appendices	II:2 A-21	M. Diaz
Baseline, set to top of box	II:1 60, 77	A. Keller
Bibliography	II:2 A-25	M. Diaz
Boxes	II:1 59, 73	A. Keller
Box numbers, automatic allocation	III:1 33	M. Plass
Branching, see If		
Capital letters		
large ~ at beginning of paragraph	II:1 60, 78	A. Keller
.	II:3 62	T _E Xarcane Class
.	II:2 A-16	M. Diaz
Roman numerals	II:1 120-121	P. Milligan, L. Price
Centering a sequence of lines	II:2 A-13	M. Diaz

Chapters and Sections	II:1 60-61, 79-81	A. Keller
	II:1 111-118	L. Price
	II:2 A-8-9, 20-22	M. Díaz
Characters, macros to produce special	II:1 57, 67-70	A. Keller
Chemical notation	II:3 57-58	M. Nichols, B. Beeth
Columns		
balanced	II:3 58-59	L. Price
multiple	II:2 A-38-40	M. Díaz
	II:3 24-25	B. Beaton
Comparison of integral values	II:1 119-120	P. Milligan, L. Price
Counters		
automatic allocation	III:1 33	M. Pless
pseudo	II:1 60, 77	A. Keller
	II:1 120	P. Milligan, L. Price
Cross references	II:3 24	B. Beaton
Deferred output	II:1 60, 66-68	A. Keller
Division	II:2 47	B. McKay
Equality of integral values	II:1 119-120	P. Milligan, L. Price
Figures	II:2 A-25-27	M. Díaz
Font		
declaring families of a particular point size	II:1 56-57, 65-66	A. Keller
	II:2 A-11	M. Díaz
definition	II:1 119	P. Milligan, L. Price
	II:2 44-45	P. Milligan
display in table form	III:1 35	R. Beaman
Footnotes	II:1 58, 71-72	A. Keller
	II:2 A-24-25	M. Díaz
French	II:2 A-12	M. Díaz
Graphics	II:2 48-49	B. McKay
	II:3 63	TjKercane Class
Headings, page	II:2 A-23-24	M. Díaz
Hidden Text	II:3 61	TjKercane Class
If		
comparison of integral values	II:1 119-120	P. Milligan, L. Price
groupless \if	II:2 46	B. McKay
null string, see Null string		
testing math-style (display, script or scriptscript)	II:2 46	B. McKay
Index production	I:1 Appendix A	T. Winograd, W. Paxton
	II:2 A-20	M. Díaz
Justification		
of reviewer's names	II:3 62	TjKercane Class
right ~	II:3 63	TjKercane Class
Letters	II:2 A-32-35	M. Díaz
Letterhead	II:2 A-33	M. Díaz
Lists	II:1 59, 72-72	A. Keller
	II:1 98-110	L. Price
	II:2 A-15	M. Díaz
Margins	II:2 A-19	M. Díaz
Matrices	II:2 A-30	M. Díaz
Memos	II:2 A-32-35	M. Díaz
Miscellaneous		
automatic printing of macro names	II:3 60-61	L. Price
avoiding "Argument of (control sequence) has an extra }."	II:2 50	M. Spivak
conditional evaluation of macros	II:2 50	M. Spivak
input-dependent macro redefinition	II:3 59-60	L. Price
\input within \if	II:2 50	M. Spivak
single tokens, identifying	II:2 52	M. Spivak
Multiplication	II:2 47	B. McKay
Null		
macros	II:1 59-60, 74-76	A. Keller
	II:2 A-16-18, 36	M. Díaz
program (SAIL)	II:1 87-93	L. Price, P. Milligan
program (Pascal)	II:1 94-97	L. Price, P. Milligan
program errata (SAIL and Pascal)	II:2 43-44	
Notes		
output to the writer on a separate file	II:1 60, 76, 85	A. Keller
printed at end of document	II:2 A-25	M. Díaz
Null string, testing for	II:1 60, 77	A. Keller
	II:2 51-52	M. Spivak
Numbering, page	II:1 57, 70-71	A. Keller
Output routines	II:1 57-58, 60-62, A. Keller	
	71, 82-85	
	II:2 A-18, 40	M. Díaz
Overlining	II:2 A-13	M. Díaz
Page numbering	II:1 57, 70-71	A. Keller
	II:2 A-18, 23	M. Díaz
Paragraphs		
beginning with large capital letters	II:1 60, 78	A. Keller
	II:2 A-16	M. Díaz
indented	II:1 58, 72	A. Keller
	II:2 A-13-15	M. Díaz
numbered, see Lists		
Parentheses, assorted sizes	II:2 A-11	M. Díaz
Pictures, plotting	II:2 48-49	B. McKay
Point, declaring font families of a particular ~ size	II:1 56-57, 65-66	A. Keller
	II:2 A-11	M. Díaz
Proofs	II:2 A-31-32	M. Díaz
Recursion	II:2 46-48	B. McKay
	II:2 53	M. Spivak
References	II:2 A-25	M. Díaz
Roman numerals, uppercase	II:1 120-121	P. Milligan, L. Price
Scaling charts	III:1 39	R. Beaman
Spanish	II:2 A-12	M. Díaz
Strings		
testing for ~ equivalence	II:3 61	L. Price
testing for the null ~	II:1 60, 77	A. Keller
	II:2 51-51	M. Spivak
Syntax charts	II:3 39-56	M. Pless
Table of Contents	II:1 60, 62, 86	A. Keller
	II:1 111-118	L. Price
	II:2 A-27-28	M. Díaz
	II:3 24	B. Beaton
Tables	II:2 A-25-27	M. Díaz
Testing		
integral values	II:1 119-120	P. Milligan, L. Price
math-style (display, script or scriptscript)	II:2 46	B. McKay
for string equivalence	II:3 61	L. Price
for the null string	II:1 60, 77	A. Keller
	II:2 51-52	M. Spivak
Theorems	II:2 A-31-32	M. Díaz
Top, baseline set to ~ of box	II:1 60, 77	A. Keller
TUGboat submissions	II:1 53-54	B. Beaton
	II:3 25	B. Beaton
Underlining	II:1 59, 73	A. Keller
	II:2 A-13	M. Díaz
Uppercase letters		
large ~ at beginning of paragraph	II:1 60, 78	A. Keller
	II:2 A-16	M. Díaz
Roman numerals	II:1 120-121	P. Milligan, L. Price

Verbatim			
mode	II:1	59-60, 74-76	A. Keller
	II:2	A-16-18, 36	M. Diaz
program (SAIL)	II:1	87-93	L. Price, P. Milligan
program (Pascal)	II:1	94-97	L. Price, P. Milligan
Vertical text	II:3	64	TgXarcane Class

* * * * *

DISPLAY OF A FONT IN TABLE FORM

Roger L. Beeman
Boeing Aerospace Company

`baselineskip` and `lineskip` are turned off to get them out of the way. `vsize` is increased to the size of my Versatec page. The output routine is redefined mostly to turn off the page numbering but `advancecount` is retained so that the page numbers displayed on the terminal will advance.

The character 0 from `cms10` is boxed so that its height and width will be available. `spike` defines an empty `vbox` which is used to assure that the horizontal rows are tall enough for the row number to fit without overflowing. `cell` is the basic box that holds one character, centered with a vertical rule on its right border. `label` uses the height of `box8` which may be different for each row and centers the octal tag rather than putting it on the same baseline as the rest of the row. The box width of 35pt is used to allay fears that the labels would not all turn out the same width and must be known later anyway. The `1em` of skip is inside the brackets and thus taken from `cms10`. `sepro` is used to add 2pts to the top and bottom of each cell.

`cellrow` saves the row of eight cells in `box8` so that `label` can use `ht8` for vertical centering. The `spike` is used to guarantee a minimum height. The height before boxing will be the maximum of this and the tallest character plus the 2pts from `sepro`. The boxing will cause a box of zero depth with the final height also including the maximum depth plus another 2pts from the second `sepro` and the height of the hrule.

This is probably the best place to point out what I really wanted was for the height above the highest character to equal the depth below the baseline. As it is, there is 2pt above the highest character and 2pt below the deepest. I probably wouldn't have given up except that `cmr30` was already pretty tight on the page and page breaking was not appetizing. Actually when it was working this well I was pretty relieved.

`1col` labels the top, again in `cms10`. `chw`, `colw` and `setw` are used to find the maximum width of any character in the font. `getw` takes the maximum over the set of characters in the font, the width of the 0 used in labeling the columns, and `1em` in the font (maybe unnecessary) then sets the variable unit to $1\frac{1}{2}$ this value. The `1vu` is used as the width of each character cell.

Finally, `table` is defined to use the given character to define the font, set the font and build the table. The `hbox` has glue to center if possible but to left justify with right overflowing forgiven if necessary. The font name is included in `cms10`. The top label and the top rule for the font cell set are followed by the sixteen cellrows.

Editor's note: The two tables which follow were pasted up from Varian copy generated at the Math Society. A few changes were necessary: new letter codes were assigned to the two fonts because of conflicts with codes already assigned to preloaded fonts; `cmr28` does not exist at the Society, so `cmr30` was substituted.

We discovered after looking at the first output that this routine neatly illuminates probable errors in a couple of METAFONT descriptions. In the `cmr30` table, row '000 has too much depth, and character '121, "Q", has no depth where one would have expected it. On checking the METAFONT descriptions, we found that the depth of the "Q" has disappeared (presumably accidentally—it was present in the original published description of the Computer Modern family), and that character '002, "Θ", has always been assigned a depth equal to that of a comma.