* * * * * * * * * * *

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`, `\dhpen` 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 COLUMN

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.

* * * * * * * * * * *

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 overfilling. `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`. `seprow` 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 `seprow`. The boxing will cause a box of zero depth with the final height also including the maximum depth plus another 2pts from the second `seprow` 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.

`lcol` 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 $1\frac{5}{8}$ 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 overfilling 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.