The Oldenburg eTeX/LaTeX3/conTeXt meeting

David Carlisle

Present
etex Peter Breitenlohner, Philip Taylor, Bernd Raichle, Jiri Zlatuska, Karel Skoupy.
latex3 Frank Mittelbach, Rainer Schöpf, Chris Rowley, Matthias Clasen, David Carlisle.
context Hans Hagen, Taco Hoekwater.

Categories
To start the discussion, Peter (initially) suggested some basic ‘categories’ of problem that could be discussed. The following discussion did not always follow this categorisation but it proved a useful starting point (in fact this list was modified during the discussion in the first session, and ended up something like this:

Syntax/Expansion/Programming Extensions to the macro programming language, not directly affecting typesetting.
Line-breaking/hmode Full typography in hmode. A switch to allow hboxes to be constructed with all the features that currently are only active in outer hmode. (Language nodes etc).
Page-breaking/vmode Discretionaries after -. Switch to turn off automatic insertion of discretionaries after (ligatures ending in) -. 
Grid Typesetting
Characters and Fonts
Alignments
Math
Character Attributes (eg Colour)
Translation (eg I/O)
List Manipulations Manipulations on token or node lists.

Expansion Control
This was more or less discussed by email before the meeting. The following new commands were agreed:

\expanded{(general text)} Expandable command returning the full expansion of the tokens in (general text).
\expandafter{(number)} One level expand the nth token ahead. (\expandafter\tw@)

The following variants were discussed with no definite conclusion (?)

Real Register A data type that has the same behaviour/arithmetic as (dimen) but without the need to supply units. (This would save the many uses of \strip@pt in latex where units are added just to do arithmetic, and then need to be removed.

Boolean Datatype Some question as to what if anything was wanted here. Boolean variables, or boolean expressions, or... Also any extension would need to fit with current \if\fi nesting behaviour.

Local/Global Assignments Some mechanism to specify that within the current scope global assignments are ‘global’ to nested groups but local to the current group.2

Line Breaking (hmode)

Full typography in hmode A switch to allow hboxes to be constructed with all the features that currently are only active in outer hmode. (Language nodes etc).

Discretionaries after - Switch to turn off automatic insertion of discretionaries after (ligatures ending in) -.

Hyphen desirability levels Two basic proposals.

1. Extended hyphenation patterns with weighted hyphenation points.
2. Extended hyphenation algorithm that tries first with patterns designed to split on compound word boundaries, then if needed a second pass with a finer set of patterns.

Notes:
1. not everyone present all the time
2. Rainer sketched a possible interface on the board, which didn’t make it to my notes — Rainer?

238 MAPS
Second version has possible advantages in keeping the current hyphenation file format. Not clear what exactly is needed.

### Page Building (vmode)

Need for a vertical analogue of \discretionary identified. (Probably just make \discretionary work in vmode.) This would seem to solve several problems with current lack of control over top-of-page behaviour.

**Agreed** new commands:

- \forcebreakpenaltylimit If set to $\geq -10000$ this is ignored, but if set to $< -10000$ is taken as the threshold at which penalties force a break.
- \nthmark\{mark class\}\{mark number\}\{box number\}
- \nummarks\{number\}\{number\}
  - \nthmark\{mark class\}\{mark number\}\{box number\} returns the number of marks of that class in the specified box.
  - \nummarks\{number\}\{number\} returns the nth mark of the specified class. (Generalises \firstmark).

This pair of commands was thought better than the original suggestion of an \allmarks command returning a list of all marks.

Other items to be considered:

- \outputpenalty10000 Look at removing the penalty node in the \outputpenalty10000 case.
- \holdinginserts \holdinginserts=2 (or different name for compatibility reasons) Hold but also copy inserts (ie a combination of current behaviour for 0 and $\geq 1$). Alternatively just use \holdinginserts=1 but give access to individual inserts in some way similar to \nthmark.
- \buildpagehook Insert a token register at the point where ‘build page’ is called. The (effective) default contents of the register would be \buildpage, a new primitive that actually builds the page. cf \output/\shipout.

### Node Handling

Perennial discussion about lack of \last* and \un* for most node types, and problems with revealing machine dependence and lack of register type corresponding to some nodes, eg rules.

Agree to look at \removelastnode.

### Parsing General Texts

There was some discussion of generalising the \uppercase mapping of character tokens, although any decision would have to wait to see the final outcome of the proposals for symbolic character handling (see below)

- **General idea** is \map\{map name\}\{general text\}. Thus \uppercase would be \map\{uccode\}(or perhaps a special csname to refer to the \uppercase map).

Some discussion of whether a token-token map would be enough (especially for the kind of parsing done in the context system) or whether a more general multiple token to multiple token system would be needed, but that leads down the road to implementing regexp replace (or OTP) and it was not clear if that fitted the etex framework.

An alternative to having an explicit map prefix command would be to have a way of automatically applying the map to input characters (but see later discussion on encoding support).

### Mathematics

Matthias Clasen reported on an investigation of possible extensions that he had recently done with Michael Downes. (See document Matthias posted.) Briefly the main points were:

- **mathstyle** The providing of an integer valued variable that reveals and controls the current math mode (would work in a way similar to \interactionmode). This would require the provision of a prefix form for \over and friends. \over would still work but would be documented as not setting the mathstyle variable to the expected value if this infix form is used.
- **\ifcramped** Test for cramped style, together with switch to force entry to cramped style.
- **kerning** Between more kinds of math atoms.
- **Spacing table** Access to the table of spaces inserted between the math atoms.
- **math classes** Extend the number of these to allow more varied automatic spacing possibilities. Also primitive to ask the math class of an atom. (This would save bm taking the \meaning and then parsing the information out of the hex string returned.)
- **subformulas** Avoid boxing subformulas.
- **glyph positions** Remove dependence on weird glyph positioning in the font, like radical hanging below baseline, so math fonts could be used with other systems.

---

3. Matthias has since experimented with some change files to implement some of these ideas, see ftp://peano.mathematik.uni-freiburg.de/pub/etex
radical extensions  Right end for radical? Or (harder)  
Both ends extending together.

Horizontal extensibles  Some discussion of whether 
this would need an extension of tfm format. Outcome 
seemed to be that it would not, just use existing tfm data 
structure, the TeX primitive would know whether it was 
constructing a horizontal or vertical extension.

overloading of font dims  Agreed to look at 
reducing this.

Fixed parameters  eg clearance around rules. Agreed to 
look at this as well.

under accents  Use information that could be coded 
in the ‘reverse’ kerning with the skewchar for underaccents. However full accent solutions, including 
multiple accents requires table driven lookup. or opt or 

16 families  Increase?

tfm  If format was extended, possible benefits?

Inner loop — lig/kern problems

After discussing several strategies for solving the problem 
that anything non expandable breaks ligature and kerning, 
including the possibility of allowing assignments in the in-
ner loop (finally thought to dangerous) the following pro-
posal (from Bernd, initially) was agreed to be worthy of 
further consideration: 
Redesign hpack routine and the linebreak algorithm so 
that kerning and ligature is separated. First create the hlist 
with no lig/kern then make a second pass adding the ligat-
ures and kerns. (This is the conceptual algorithm, an im-
plementation may want to interleave these processes when 
’safe’ eg word by word.)

This would mean that fi always produced the fi ligature. 
new primitive would be needed to break ligatures but not 
break hyphenation process also similarly \nokern.

Some questions on semantics of unhbox in this setup, 
whether the list should be “re-ligatured”.

Modified algorithm could further split things up separ-
ating ligatures first then kerning. (Currently tex just does 
whichever comes first in the fonts lig/kern table if both are 
specified for a pair)

Also discussed mechanisms of overriding ligatures for a 
font eg the Portugese no fi ligature example. Also, simpler 
the possibility to turn off all ligatures in a font, for verbatim 
type uses.

Reconsider paragraph

If the possibility is given to re-break an hlist, tex needs 
to return more information about the resulting paragraph to 
enable a choice to be programmed. (It was thought unlikely 
that one could algorithmically set the paragraph parameters 
to ‘correct’ an earlier try, but one could at least try several 
pre-programmed possibilities, and then use the returned in-
formation to choose the ‘best’ outcome.

Possible info that might be useful

- Information currently only available in display math \(\text{\texttt{\backslash predisplaysize}}\)
- total number of hyphens
- maximum number of adjacent hyphens
- maximum badness of a line
- minimum badness of a line
- total demerits
- fit class of worst line

The plan implemented but not released in V2 broke the 
paragraph onto the current vertical list, as normal but re-
turned an hlist for further tries. This means that you need to 
program the removal of the original paragraph if you need to 
retry, which means not working on the main vertical list. Some alternatives were considered.

Basic scheme in TeX is:
construct hlist until reach $$ or \texttt{\textbackslash par} or end-of-vmode-
group. Then

- Modification at end (unskip, add parfillskip etc), leave 
hmode.
- (A) look at legal breakpoints & choose optimal path (+/- \texttt{\textbackslash looseness})
- 
line break decision (1st pass, hyph, 2nd pass, 3rd pass)
- (B) 
postlinebreak (package, migration, append to vlist)
- in vmode or dispaly math (first point tex looks for token)

Plan 1  copy list at (A) into one global variable (destroy 
previous contents). This is a private unpackaged hbox.
New primitive to ‘unhbox’ this where required.

Plan 2  At (B) call ‘output routine’. Has badness 
information from linebreaking routine but not vertical 
size information. (This plan rejected, as vertical size 
information seems vital for intended applications).

Plan 2b Append the new vertical material, but don’t 
exercise the page builder.

Word Grabbing

The following proposal seemed the most plausible (after 
several other schemes were rejected)

New expandable primitive \texttt{\textbackslash grabword} 
Expands tokens up to the first non expandable token that 
is not a \texttt{(letter-other-space)}. Returns the resulting tokens 
surrounded by explicit \texttt{\{\}} tokens and leaves the ‘terminat-
ing’ non expandable token in place.
Typical use
\def\foo#1{<pcdata>#1</pcdata>}
\expandafter\foo\grabword user supplied stuff.

Note that the ‘word’ that is picked up will be prematurely terminated by all the stuff that currently breaks ligatures, but if the symbolic character proposals could be implemented these would rarely need to appear mid word.

Named Reference Points
Named reference points on (h or v)boxes.

Like \mark\specialetc special named reference point into current (vh)list.

When hbox is packaged reference point node has a position relative to reference point on box (integer arithmetic).

These positions could be made available similar to current \wd and friends.

\xxxxx
\number\name

takes a box number and a symbolic name and returns the position (might need two commands to get x,y coordinates, or one that returns a pair)

When unbox a vertical list the information is lost but the reference nodes are still in the list so their positions in the new list will be made available when the new list is packaged.

When packaging an outer box, inherit all labels from inner ones.

Possibly (or more likely not) return some some info about the nesting.

Would the linebreaking algorithm need to add labels to each line???

Special \specials
A \special for writing positional information to the dvi file?
A delayed \special. (cf non-immediate \write).

Combining penalties
\penalty-500 \penalty10000 currently still allowable break, so user has no way to overrule some automatically inserted penalties. Various algorithms considered, eg some average, max, min, etc. In practice probably doesn’t matter as long as it preserves ‘finiteness’.

Any algorithm that allows to overwrite existing penalties that favour a break would be a very big improvement and should be looked at closely.

Symbolic Characters
Some schematics of current and proposed flow of ‘characters’ through tex. (Peter can write on the board faster than I can draw ascii art on this laptop. The following diagrams probably are rubbish, but I keep them exactly for now. Need replacing by something a bit more meaningful)\footnote{Mathias has made some more extensive notes on this subject}

\begin{verbatim}
1:1 font encoding
input file --> char tokens ---> char nodes --> dvi nos
\downarrow \downarrow
write

want:

symbolic char
\textbackslash

1:1 font encoding
input file --> char tokens ---> char nodes --> dvi nos
\downarrow \downarrow
write

\end{verbatim}

for each language have subset \( \leq 256 \) SC which map to hyphenable ascii codes

SC+language encoding+font encoding ————–

Hyphenation Patterns (always based on 8bit table)
Possibly the mapping from symbolic characters to ascii codes may be many to one?
SC + font enc — represented in a font (somehow) possibility of using unicode as ‘symbolic names’

[This section needs filling in in a lot more detail]

Parameter matching
Two sets of extensions to parameter matching were considered.

Alternative delimited argument tokens
\def\foo#1\a#2\b#3\c\d{...}

The argument #1 would be all the tokens up to the first occurrence at the current brace group level of \a or \b or \c.

Within the definition the following forms would be available

#1 the tokens of the first argument.

#2 the tokens of the second (non-delimited) argument.
As the second argument is not delimited it is not clear what this should be. Probably empty, but could be an error, either would be OK as a behaviour.

Note that this extension is upwords compatible as the forms are all error conditions in the current tex.

A step towards regexp matching
Currently \def\foo#1#2#3\xx{} always assigns the smallest possible arguments to the first and second arguments with the third argument taking up the maximum number of tokens up to the delimiting token.

Proposal to allow
\def\foo#*1#2#?3\xx{}

#*1 argument may take non or more brace groups or tokens.

#*1 argument may take one or more brace groups or tokens.

#?1 argument may take at most one brace groups or tokens.

February 26–28 1998