

# The `latex-lab-math` code\*

Frank Mittelbach, Joseph Wright, L<sup>A</sup>T<sub>E</sub>X Project

v0.5i 2024-02-09

## Abstract

This is an experimental prototype. It captures math material (basically okay, but the interfaces for packages aren't yet there) and tags the material (which is not yet anywhere near the final state). That part is provided for experimentation and to gather feedback, etc.

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Math capture</b>	<b>3</b>
	2.1 Code level interfaces . . . . .	3
	2.2 Document level interfaces . . . . .	3
<b>3</b>	<b>Math tagging</b>	<b>3</b>
	3.1 Inline math . . . . .	4
	3.2 Display math . . . . .	5
	3.3 Associated Files . . . . .	5
	3.4 Options . . . . .	6
<b>4</b>	<b>Known current bugs, etc.</b>	<b>6</b>
	4.1 Capture/grabbing problems . . . . .	6
	4.2 Fake math . . . . .	7
	4.3 Processor . . . . .	7
	4.4 Other problems . . . . .	7
	4.5 Other Todos . . . . .	8

---

\*

<b>5</b>	<b>The Implementation</b>	<b>8</b>
5.1	File declaration	8
5.2	Setup	8
5.3	Data structures	9
5.4	Tagging tools	9
5.5	Code related to AF	10
5.6	Mathstyle detection	13
5.7	Tagging options	14
5.8	Sockets	14
5.8.1	Main inline math sockets	14
5.8.2	Main display math sockets	15
5.8.3	Internal sockets	16
5.9	Interface commands	21
5.10	Content grabbing	22
5.11	Marking math environments	23
5.12	Document commands	27
5.13	<code>\everymath</code> and <code>\everydisplay</code>	29
5.14	Modifying kernel environments	29
5.15	Disable math grabbing in the <code>begindocument</code> hook	30
5.16	Modifying <code>amsmath</code>	30

<b>Index</b>	<b>36</b>
--------------	-----------

## 1 Introduction

Todo: update all the documentation! Both here and (what little there is!) in the implementation section.

Tagging math involves a variety of tasks that require that math is captured before the typesetting

- When typesetting the math MC-tags and structure commands must be inserted at the begin and the end, and perhaps also around lines or other subparts of the equation.
- The source and/or a mathml-representation of the source must be available so that it can be (perhaps after some preprocessing) be used in an associated file or in an alternate text
- It must be possible to measure the math for e.g. a `bbbox` setting.

This file implements capture of all math mode material at the outer level, i.e., a formula is captured in its entirety with inner text blocks (possibly containing further math) absorbed as part of the formula. For example,

```
\[ a \in A \text{ for all } a<5\$ \]
```

would only result in a single capture of the tokens “`a\in A\text{for all }a<5\$`”.

## 2 Math capture

In the current setup

- $\$, \backslash(\dots\backslash)$  and  $\$\$$  grab (through a command in `\everymath/cseverydisplay`) if the boolean `\l_@@_collected_bool` is false. If the boolean is true they behave normally and can for example contain verbatim.
- All (registered) environments grab their body regardless of the state of the boolean. For `equation`, `equation*` and `math` this is a change as they no longer can contain verbatim.
- BUG: `\[...\]` grabs if `\l_@@_collected_bool` is false. If it is true it falls back to `equation*` and then errors because this can't find the end.

### 2.1 Code level interfaces

---

```
\math_register_env:n \math_register_env:n {<env>}  
\math_register_env:nn \math_register_env:nn {<env>} {<options>}
```

---

Registers the `<env>` as a math environment which should be captured and made available. This is necessary for all top-level math mode environments: low-level errors may result if these are not correct set up. One or more key-value `<options>` may also be given:

**arg-spec** The argument specification taken by the beginning of the environment; this is used to remove non-mathematical material.

---

```
\math_processor:n \math_processor:n {<tokens>}
```

---

Declares that the captured math content should be passed to the `<tokens>`, which will receive the environment type as `#1` and the content as `#2`. The processing is done before the typesetting. It is not applied if `\ifmeasuring@` is true.

### 2.2 Document level interfaces

---

```
\RegisterMathEnvironment \RegisterMathEnvironment [(<options>)] {<env>}
```

---

Registers the `<env>` as a math environment which should be captured and made available. This is necessary for all top-level math mode environments: low-level errors may result if these are not correct set up. One or more key-value `<options>` may also be given:

**arg-spec** The argument specification taken by the beginning of the environment; this is used to remove non-mathematical material.

## 3 Math tagging

The tagging code has to handle

- the embedding into the surrounding. This means
  - closing and reopening MC-chunks

- closing and reopening text/P-structures
- handling interferences of the tagging code with penalties and spacing.
- the actual tagging which means to do some or all of the following tasks:
  - setup content for an associated source file
  - setup content for an associated mathml file
  - setup content for the /Alt key
  - setup content for the /ActualText key
  - setup attributes
  - add associated files
  - add a Formula structure
  - surround subparts (e.g. lines) with Formula sub structures (perhaps with their own set of additional content)
  - surround elements of the equation with mathml structure elements (currently only luatex with luamml)

### 3.1 Inline math

The embedding code is added through the sockets

- `tagsupport/math/inline/begin`
- `tagsupport/math/inline/end`

The sockets simply push and pop the MC currently. Without tagging they use the noop-plug.

The actual tagging is in done through the sockets

- `tagsupport/math/inline/formula/begin` This socket takes the math as argument and its code should output it for typesetting. Without tagging the socket uses the identity plug. The default plug of the socket calls these three internal sockets for the tagging support:
  - `tagsupport/math/content` This should set up the various content variables (empty variables are ignored by the structure code and so can be used to suppress a setting).
  - `tagsupport/math/struct/begin` This calls `\tag_struct_begin:n`. It should also write the associated files if needed.
  - `tagsupport/math/substruct/begin` this handles subparts. TODO: does it really make sense in inline math to have that??
- `tagsupport/math/inline/formula/end` This socket ends the formula structure(s). The default plug calls these internal sockets:
  - `tagsupport/math/substruct/end`
  - `tagsupport/math/struct/end`

## 3.2 Display math

*to be written*

## 3.3 Associated Files

The current code allows the attachment of two types of associated file to the Formula structure: the L<sup>A</sup>T<sub>E</sub>X source and a MathML representation. Technically both can be attached—AF is an array of file references—practically there can be problems with PDF consumers: e.g. ngpdf used both and so showed the equation twice (this has been corrected in the newest version) and Foxit seems to see only the first AF in the array (so we attach the mathml as first file).

The L<sup>A</sup>T<sub>E</sub>X source can be (and is) attached automatically. It can be suppressed by an option with `math/tex/AF=false`, see below.

The MathML is attached if a file `\jobname-mathml.html` is found and if it contains a suitable MathML snippet for the current formula. The MathML processing can be suppressed globally by emptying the list of mathml files with `math/mathml/sources=`. Locally for a formula `math/mathml/AF=false` can be used.

For a MathML representation a file with such representations must be provided. If the equation is numbered the numbering should be part of the MathML as the L<sub>b</sub>l substructure is ignored if an MathML is used (see [https://github.com/foxitsoftware/PDF\\_UA-2](https://github.com/foxitsoftware/PDF_UA-2)).

The MathML representation is given in a special format. It is meant to be a valid html file that can be viewed in a browser. For this it can start with `<!DOCTYPE html><html>` and end with `</html>` It should have the extension `.html`. The `<mathml>` content is read with special catcodes, so can contain ambersands, hashes, comment chars and unmatched braces such as `<mo>{</mo>`

The file should contain a number of representations in this format:

```
<div>
  <h2>\mml {key}</h2>
  <p>{source}</p>
  <p>{hash}</p>
  <math {attributes} >
    {mathml}
  </math>
</div>
```

The keywords `<div>`, `<h2>\mml`, `<p>`, `<math`, `</math>` `</div>` are required as they are used to delimit the arguments by the L<sup>A</sup>T<sub>E</sub>X code.

`<key>` and `<source>` are only used for debugging, they help to identify the equation referred by this representation. The source should be used correctly escaped `&` and `<` so that it gives valid html!

`<attributes>` is not required either, but can e.g. contain attributes to improve the display in a browser:

```
<math alttext="\mathbf{G}" class="ltx_Math" display="inline">
```

It can also contain the name space declaration: `xmlns="http://www.w3.org/1998/Math/MathML"`

<sup>1</sup>

---

<sup>1</sup>But it is probably not needed and only blows up the PDF.

By default the code tries at the begin of the document to read a file `\jobname-mathml.html` in the `html`-format. The file name can be changed with `mathml/setfiles={filename1,filename2}` (without extension, `html` is added automatically). If there is a list, all files are loaded. If a file doesn't exist it is ignored, only an info is written to the log.

Currently every MathML-snippet from a file is embedded into the PDF, it is not checked first if it is actually used (simply writing everything to the PDF is a bit easier than keeping everything in memory and also means that the snippets are one after the other in the PDF).

As mentioned above the MathML-AF can be suppressed for the equations in a group with `math/mathml/AF=false`, or completely by setting `math/mathml/sources=` in the preamble.

Files embedded in a PDF can be listed in the attachments panel of a PDF viewer. This is probably not so useful for lots of small files (but one could create collections), but as long as PDF editors or viewers don't offer proper support to access the AF it can help so have them there. The MathML are added by default, but the  $\LaTeX$  source not. This can be changed with `viewer/pane/mathsource=true` (anywhere in the document) and `viewer/pane/mathml=false` (in the preamble, before the external file is read).

### 3.4 Options

## 4 Known current bugs, etc.

New Section, now with subsections.  
As indicated, these lists are probably incomplete.  
Some of these have been addressed in a more recent branch.

### 4.1 Capture/grabbing problems

1. Incorrect grabbing of  $\$$ -math when there is also explicit  $\$$ -math within a *text environment* that is itself within the math that should all be grabbed. For example,

$$\$a\begin{minipage}{1cm}\$b\end{minipage}\$$$

would only result in the capture of the tokens “`a\begin_{minipage}{1cm}`”. This can be avoided by an additional brace group:

$$\${a\begin{minipage}{1cm}\$b\end{minipage}}\$$$

2. Similar incorrect grabbing with  $\$\$$  also.
3. The grabbing, for all the display environments (and  $\backslash$   $\backslash$ ), needs to deal with nesting: `amsmath` contains code for this.
4. The math can't contain verbatim and verbatim-like commands. This is nothing new for the `amsmath` environments but changes  $\$$  and  $\backslash$ [ $\backslash$ ] and `equation*` (see e.g. tagging-project issue #30).
5. Begin and end of the math or math environment can not be hidden in commands. For example `>{\$}1<{\$}` in a tabular would lead to errors. Defining  $\backslash$ [ to fall back to `equation*` doesn't work if `equation*` is a grabbing environment.
6. The behaviour of  $\backslash$ [... $\backslash$ ] is faulty. See above.

## 4.2 Fake math

In a number of places in L<sup>A</sup>T<sub>E</sub>X math commands (mainly  $) is used only for technical reason, e.g. to access a math font, to setup a symbol or to use `\vcenter`.$

The code identifies such fake math mostly by making use of the `\m@th` command where two methods are used for the automatic detection:

- After grabbing math content the code checks if the content contains the token `\m@th` and if yes it doesn't call the processor before reinserting the content and perhaps adding tagging code. This method requires that the math can be grabbed (e.g. that the end dollar is visible) and that the `\m@th` is visible. It applies for example in `\@iiiparbox` where the code from `$$\vcenter` to `\m@th$` is grabbed and put back. It does not work for example for `tabular` where the dollars and the `\m@th` token are spread around over three commands. `tabular` needs therefore manual intervention. A look in the list of usages (in `usage-of-m@th.md`) justifies this approach. All usages are either not math at all, or related to small elements that probably shouldn't be grabbed and processed on their own.
- `\m@th` is redefined so that it sets the boolean `\l_@@_collected_bool` to true. If `\m@th` is used inside math that has been grabbed this doesn't change much as the boolean is set by the grabbing anyway. For usages outside math the benefit is not so clear: The setting avoids that in L<sup>A</sup>T<sub>E</sub>X<sub>ε</sub> the epsilon is processed as math, but it also prevents that the content of the `amsmath` command `\boxed` is processed as math. It means that if one wants to reenable math processing inside some (fake) math one has to do it after `\m@th` calls.

Open problems

1. The grabbing code doesn't pass the info that it detected a `\m@th` token. This means that the tagging code has to do the same check (and doesn't do this in all cases yet).
2. Commands are missing to locally disable the grabbing and processing, e.g. to handle `tabular`.
3. It must be checked if setting the boolean in `\m@th` really makes sense or if commands like L<sup>A</sup>T<sub>E</sub>X<sub>ε</sub> should be handled manually.

## 4.3 Processor

The grabbed math is at first passed to the processor. The processor is not called in a measuring phase (from the `amsmath \ifmeasuring@`) and if the `\m@th` token is detected. It is not quite clear what purpose the processor has. As it is a public interface it can't be used for internal code. And typesetting happens later and the processor can't really change this. Currently it is mostly used for debugging and messages. If the `\m@th` is found the `\l_@@_fakemath_bool` is set, so if the code is changed this must be preserved.

## 4.4 Other problems

1. The presence of `\m@th` in association with `\ensuremath` does not necessarily indicate fakemath. This is because wanting `mathsurround` to be zero is very reasonable and common, *even when the math is genuine* (and hence needs to be collected).  
TODO: this claim needs some examples.

2. User-defined environments can create problems; but this area, of new, copied and changed environments, has not yet been developed.

Joseph wrote, inter alia:  
 My thinking [regarding] `\RegisterMathEnvironment`  
 - (New) Math environments should not be created-then-patched, but only generated by a [(future)] dedicated command (`\DeclareMathEnvironment`, presumably)  
 - Math environments created with `ltxcmd` [commands] should not be copied, . . .  
 - Package authors should be able to manually set up math environments with a public boolean.

## 4.5 Other ToDos

1. Add (some of) the math display commands that were “lifted from plain”, e.g., `\displaylines` `\eqalign{??}`.
2. The `breqn` packages changes catcodes and that isn’t yet covered by our mechanism.
3. `\intertext` is not correctly taken into account by the code splitting multiline math into subformulas.

`\MaybeStop` (temporarily) not executed, as it is unknown on Chris’ system.

## 5 The Implementation

```
1 <@=math>
2 <*kernel>
```

### 5.1 File declaration

```
3 \ProvidesFile{latex-lab-math.ltx}
4     [\ltxlabmathdate\space
5     v\ltxlabmathversion\space
6     Grab all the math(s) and tag it (experiments)]
7
8 Temp loading ...
9 \AddToHook{begindocument/before}{\RequirePackage{latex-lab-testphase-block}}
10 \ExplSyntaxOn
```

Change description here?

### 5.2 Setup

Loading `amsmath` is an absolute requirement: this avoids needing to have conditional definitions and deals with how to define `\[/\]` neatly.

```
9 \AddToHook{begindocument/before}{ \RequirePackage { amsmath } }
```



### 5.3 Data structures

---

`\l__math_collected_bool` Tracks whether math mode material has been collected, which happens inside `amsmath` environments as well as those handled directly here. If true following math will not grab and/or process. See 2 for details.

```
10 \bool_new:N \l__math_collected_bool
```

---

`\l__math_fakemath_bool` Tracks whether math mode material has been identified as fake math during the grabbing phase, which happens currently if the grabbed contents contains the `\m@th` token.

```
11 \bool_new:N \l__math_fakemath_bool
```

Change first tl name below: 'env' => 'info'?

Or do we need an extra

`\g__math_grabbed_env_tl`

`\g__math_grabbed_math_tl`

`\g__math_grabbed_env_tl` contains the name of the math environment (`math` in the case of inline math, `\g__math_grabbed_math_tl` the math content.

```
12 \tl_new:N \g__math_grabbed_env_tl
```

```
13 \tl_new:N \g__math_grabbed_math_tl
```

---

`\l__math_tmpa_tl` Temporary variables

`\l__math_tmpa_skip`

`\l__math_tmpa_str`

```
14 \tl_new:N \l__math_tmpa_tl
```

```
15 \skip_new:N \l__math_tmpa_skip
```

---

`\l__math_content_alt_tl` Temporary variables to hold math content that should be used in actual or alt text and stored as AF.

`\l__math_content_actual_tl`

`\l__math_content_AF_tl`

```
16 \tl_new:N \l__math_content_alt_tl
```

```
17 \tl_new:N \l__math_content_actual_tl
```

```
18 \tl_new:N \l__math_content_AF_source_tl
```

```
19 \tl_new:N \l__math_content_AF_source_tmpa_tl
```

```
20 \tl_new:N \l__math_content_AF_mathml_tl
```

### 5.4 Tagging tools

The following commands implement small tagging code chunks. This should probably be collected and moved into `tagpdf` later.

`\__tag_tool_close_P:` This closes a P/text-chunk, both the MC and the structure and increases the counter manually.

```
21 \cs_new_protected:Npn \__tag_tool_close_P:
22 {
23   \tag_if_active:T
24   {
25     \tag_mc_end: %end P-chunk, should perhaps be \tag_mc_end_push: ...
26     \__tag_gincr_para_end_int:
27     \__tag_check_para_end_show:nn{red}{} %debug: show para
```

```

28     \tag_struct_end:
29     }
30 }

```

(End of definition for `\_tag_tool_close_P:`)

We add also an attribute.

```

31 \tl_new:N\l__math_attribute_class_tl
32 \tagpdfsetup
33     {newattribute = {inline}      {/0 /Layout /Placement/Inline},
34     newattribute = {display}     {/0 /Layout /Placement/Block},
35     }

```

## 5.5 Code related to AF

Booleans to handle the options.

---

```

\l__tag_math_texsource_AF_bool
\l__tag_math_texsource_pane_bool
\l__tag_math_mathml_AF_bool
\g__tag_math_mathml_AF_bool
\l__tag_math_mathml_pane_bool
\l__tag_math_alt_bool

```

---

```

36 \bool_new:N\l__tag_math_texsource_AF_bool
37 \bool_new:N\l__tag_math_texsource_pane_bool
38 \bool_new:N\l__tag_math_mathml_AF_bool
39 \bool_new:N\g__tag_math_mathml_AF_bool
40 \bool_new:N\l__tag_math_mathml_pane_bool
41 \bool_new:N\l__tag_math_alt_bool

```

---

```

\g__math_mathml_total_int
\g__math_mathml_int
\g__math_math_total_int
\g__math_mathml_AF_found_int
\g__math_mathml_AF_attached_int

```

---

`\g__math_mml_total_int` records the mathml fragments read in. `\g__math_mml_int` records the mathml fragments read in with a different hash. `\g__math_AF_total_int` records the number of math structures that try to attach a mathml AF. `\g__math_AF_found_int` records the number of math structures for which a fitting mathml is found. `\g__math_AF_attached_int` records the number of math structures which got a mathml fragment (if mathml-AF are not disabled locally this should be the equal to the previous number).

```

42 \int_new:N\g__math_mathml_total_int
43 \int_new:N\g__math_mathml_int
44 \int_new:N\g__math_math_total_int
45 \int_new:N\g__math_mathml_AF_found_int
46 \int_new:N\g__math_mathml_AF_attached_int

```

---

---

`\l__tag_math_mathml_files_clist`

A sequence to store the file list for the mathml.

```
47 \clist_new:N\l__tag_math_mathml_files_clist
48 \clist_put_right:Ne\l__tag_math_mathml_files_clist {\c_sys_jobname_str-mathml}
```

This is the internal variant of the `\mml` command.

`\__math_AF_mml:nnnn`

```
49 \cs_new_protected:Npn \__math_AF_mml:nnnn #1 #2 #3 #4
50 % #1 number, #2 tex source for debugging, #3 hash, #4 mathml
```

```
51 {
52   \int_gincr:N \g__math_mathml_total_int
```

mathml with the same hash should be included only once:

```
53   \tl_if_exist:cF { g__math_mathml_#3_tl }
54   {
55     \int_gincr:N \g__math_mathml_int
```

a simple Desc key, take care that it is a valid string!

```
56     \pdfdict_put:nne {l_pdffile/Filespec} {Desc}{(mathml-#1)}
57     \pdffile_embed_stream:nnN {#4}{mathml-#1.xml}\l_tmpa_tl
```

not strictly necessary but makes the files visible in the file attachment page

```
58     \bool_if:NT \l__tag_math_mathml_pane_bool
59     {\pdfmanagement_add:nne {Catalog/Names}{EmbeddedFiles}{\l_tmpa_tl}}
60     \tl_new:c{g__math_mathml_#3_tl}
61     \tl_gset_eq:cN{g__math_mathml_#3_tl}\l_tmpa_tl
62   }
63 }
```

(End of definition for `\__math_AF_mml:nnnn`.)

The html reader.

```
64 \cs_new_protected:Npn \__math_AF_html_reader:w#1</h2>#2<p>#3</p>#4<p>#5</p>#6<math{
65   \begingroup
66   \char_set_catcode_other:N\{
67   \char_set_catcode_other:N\}
68   \char_set_catcode_other:N\#
69   \char_set_catcode_other:N\%
70   \__math_AF_html_reader_verb:w{#1}{#3}{#5}<math
71 }
```

```
72 \cs_new_protected:Npn \__math_AF_html_reader_verb:w#1#2#3#4~</div>{
73   \endgroup
74   \__math_AF_mml:nnnn{#1}{#2}{#3}{#4}
75 }
```

`port/math/mathml/write (socket)` This writes a html-dummy with the hash and the math content. This should be optional, so it uses a socket that can be disabled

```
76 \socket_new:nn {tagsupport/math/mathml/write}{0}
```

On (*plug*)

```
77 \socket_new_plug:nnn{tagsupport/math/mathml/write}{On}
78 {
79   \str_set:NV\l__math_tmpa_str\l__math_content_AF_source_tl
80   \str_replace_all:Nnn\l__math_tmpa_str{&}{&amp;}
```

```

81 \str_replace_all:Nnn\l__math_tmpa_str{<}{&lt;}
82 \iow_now:Ne \g__math_writedummy_iow
83 {
84   \iow_newline:
85   <div>
86   \iow_newline:
87   <h2>\c_backslash_str mml\c_space_tl \int_use:N \g__math_math_total_int </h2>
88   \iow_newline:
89   <p>\l__math_tmpa_str</p>
90   \iow_newline:
91   <p>\l__math_content_hash_tl </p>
92   \iow_newline:
93   <math></math>
94   \iow_newline:
95   </div>
96   \iow_newline:
97   }
98 }

```

And now a key to activate the socket.

```

99
100 \keys_define:nn { __tag / setup }
101 {
102   math/mathml/write-dummy .code:n =
103   {
104     \bool_gset_true:N \g__tag_math_mathml_AF_bool
105     \tl_if_exist:NF\g__math_writedummy_iow
106     {
107       \iow_new:N \g__math_writedummy_iow
108       \iow_open:Nn \g__math_writedummy_iow
109       {
110         \c_sys_jobname_str-mathml-dummy.html
111       }
112       \iow_now:Ne \g__math_writedummy_iow
113       {
114         <!DOCTYPE~html>
115         \iow_newline:
116         <html>
117       }
118       \AssignSocketPlug {tagsupport/math/mathml/write}{On}
119       \AddToHook{enddocument/afterlastpage}
120       {
121         \iow_now:Nn \g__math_writedummy_iow {</html>}
122         \iow_close:N \g__math_writedummy_iow
123       }
124     }
125   },
126   math/mathml/write-dummy .usage:n=preamble
127 }

```

\\_math\_AF\_process\_mathml\_files:

```

128 \box_new:N\l__math_tmpa_box
129 \cs_new_protected:Npn \_math_AF_process_mathml_files:
130 {

```

```

131 \hbox_set:Nn \l__math_tmpa_box
132 {
133   \pdfdict_put:nnn { l_pdffile/Filespec }{AFRelationship} { /Supplement }
134   \pdfdict_put:nne
135     { l_pdffile }{Subtype}
136     { \pdf_name_from_unicode_e:n{application/mathml+xml} }
137   \char_set_catcode_other:N \#
138   \cs_set_eq:NN\mml \__math_AF_html_reader:w
139   \clist_map_inline:Nn \l__tag_math_mathml_files_clist
140     {
141       \file_if_exist:nTF {##1.html}
142         {
143           \typeout{Info:~reading~mathml~file~##1}
144           \file_input:n {##1.html}
145           \bool_gset_true:N\g__tag_math_mathml_AF_bool
146         }
147         {
148           \typeout{Info:~mathml~file~##1~does~not~exist}%info message
149         }
150       }
151     }
152   \bool_if:NT\g__tag_math_mathml_AF_bool
153     {
154       \typeout{Info:~Activating~mathml~support}
155       \AssignSocketPlug{tagsupport/math/struct/begin}{test-mathml}
156       \AssignSocketPlug{tagsupport/math/struct/end}{test-mathml}

```

mathml handling doesn't like subparts, so we disable them for now:

```

157       \AssignSocketPlug{tagsupport/math/substruct/begin}{single}
158       \AssignSocketPlug{tagsupport/math/substruct/end}{single}
159       \AddToHook{enddocument/info}
160         {
161           \iow_term:n{MathML~statistic}
162           \iow_term:n{=====}
163           \iow_term:e{==>~\int_use:N\g__math_mathml_total_int\c_space_tl
164             MathML~fragments~read}
165           \iow_term:e{==>~\int_use:N\g__math_mathml_int\c_space_tl
166             different~MathML~fragments}
167           \iow_term:e{==>~\int_use:N\g__math_math_total_int\c_space_tl
168             math~fragments~found}
169           \iow_term:e{==>~\int_use:N\g__math_mathml_AF_found_int\c_space_tl
170             fitting~MathML~AF~found}
171           \iow_term:e{==>~\int_use:N\g__math_mathml_AF_attached_int\c_space_tl
172             MathML~AF~attached}
173         }
174       }
175     }
176   \AddToHook{begindocument}{\__math_AF_process_mathml_files:}

```

*(End of definition for \\_\_math\_AF\_process\_mathml\_files:.)*

## 5.6 Mathstyle detection

In some cases we need to detect the mathstyle used in a `\mathchoice` command and to disable/enable tagging in the unused branches. This is currently only used in the

amstext command `\text` but is perhaps also needed in other cases, so we create a general command.

```

\l__math_mathstyle_int
\g__math_mathchoice_int
mathstyle
177 \int_new:N \l__math_mathstyle_int
178 \int_new:N \g__math_mathchoice_int
179 \property_new:nnnn{mathstyle}{now}{-1}{\int_use:N \l__math_mathstyle_int }

```

(End of definition for `\l__math_mathstyle_int`, `\g__math_mathchoice_int`, and `mathstyle`. This function is documented on page ??.)

For now internal, but perhaps will need a public version. The command should be used in every branch of a `\mathchoice` (with the correct `mathstyle` number) and with an unique label (which should be the same in every branch). `\g__math_mathchoice_int` can be e.g. increased before the `mathchoice` and then used.

```

\__math_tag_if_mathstyle:nn
180 \cs_new_protected:Npn \__math_tag_if_mathstyle:nn #1 #2
181   % #1 refers to label
182   % #2 is a number for the mathstyle (typically 0,2,4,6)
183   {
184     \int_set:Nn \l__math_mathstyle_int {#2}
185     \property_record:nn {#1} { mathstyle }
186     \int_compare:nNnTF { \property_ref:nn {#1}{ mathstyle} } = { #2 }
187       { \tag_start: } { \tag_stop: }
188   }
189 \cs_generate_variant:Nn \__math_tag_if_mathstyle:nn {en}

```

(End of definition for `\__math_tag_if_mathstyle:nn`.)

## 5.7 Tagging options

```

190 \keys_define:nn { __tag / setup }
191   {
192     math/mathml/sources .clist_set:N = \l__tag_math_mathml_files_clist,
193     math/alt/use .bool_set:N = \l__tag_math_alt_bool,
194     viewer/pane/mathml .bool_set:N = \l__tag_math_mathml_pane_bool,
195     viewer/pane/mathml .initial:n = true,
196     viewer/pane/mathsource .bool_set:N = \l__tag_math_texsource_pane_bool,
197     math/mathml/AF .bool_set:N = \l__tag_math_mathml_AF_bool,
198     math/mathml/AF .initial:n = true,
199     math/tex/AF .bool_set:N = \l__tag_math_texsource_AF_bool,
200     math/tex/AF .initial:n = true
201   }

```

## 5.8 Sockets

### 5.8.1 Main inline math sockets

`\support/math/inline/begin (socket)` The first two sockets are meant to embed inline math into the surrounding (so to close/reopen e.g. MC-chunks). The other two implement the actual formula structure.

`\support/math/inline/end (socket)`

`\math/inline/formula/begin (socket)` The formula sockets are despite their naming not symmetric: the `begin` socket is issued after the math has started, while the `end` socket is after the math!

`\math/inline/formula/end (socket)`

```

202 \socket_new:nn {tagsupport/math/inline/begin}{0}
203 \socket_new:nn {tagsupport/math/inline/end}{0}
204 \socket_new:nn {tagsupport/math/inline/formula/begin}{1} %
205 \socket_new:nn {tagsupport/math/inline/formula/end}{0}

```

MC (*plug*)

```
206 \socket_new_plug:nnn
207   {tagsupport/math/inline/begin}
208   {MC}
209   {\tag_mc_end_push:}
210 \socket_new_plug:nnn
211   {tagsupport/math/inline/end}
212   {MC}
213   {\tag_mc_begin_pop:n{}}
```

We probably will want to test different tagging recipes.

default (*plug*)

```
214 \socket_new_plug:nnn
215   {tagsupport/math/inline/formula/begin}
216   {default}
217   {
218     \socket_use:n{tagsupport/math/content}
219     \socket_use:n{tagsupport/math/struct/begin}
```

TODO: does inline math need subformula handling?

```
220     % inner formula if multiple parts (not really implemented yet)
221     \socket_use:n{tagsupport/math/substruct/begin}
222     #1
223     \socket_use:n{tagsupport/math/end}
224   }
225 \socket_new_plug:nnn
226   {tagsupport/math/inline/formula/end}
227   {default}
228   {
229     \socket_use:n{tagsupport/math/substruct/end}
230     \socket_use:n{tagsupport/math/struct/end}
231   }
```

## 5.8.2 Main display math sockets

`port/math/display/begin` (*socket*) The first two sockets are meant to embed display math into the surrounding (so to  
`support/math/display/end` (*socket*) close/reopen e.g. MC-chunks and P-structure). The other two implement the actual  
`/display/formula/begin` (*socket*) formula structure. The formula sockets are despite their naming not symmetric: the  
`math/display/formula/end` (*socket*) begin socket is issued after the math has started, while the end socket is after the math!

```
232 \socket_new:nn {tagsupport/math/display/begin}{0}
233 \socket_new:nn {tagsupport/math/display/end}{0}
234 \socket_new:nn {tagsupport/math/display/formula/begin}{1} %
235 \socket_new:nn {tagsupport/math/display/formula/end}{0}
```

default (*plug*)

```
236 \socket_new_plug:nnn
237   {tagsupport/math/display/begin}
238   {default}
239   { \_tag_tool_close_P: }
240 \socket_new_plug:nnn
241   {tagsupport/math/display/end}
242   {default}
243   {
```

```

244 }
default (plug)
245 \socket_new_plug:nmn
246 {tagsupport/math/display/formula/begin}
247 {default}
248 {
249   \socket_use:n{tagsupport/math/content}
250   \socket_use:n{tagsupport/math/struct/begin}
251   \socket_use:n{tagsupport/math/substruct/begin}
252   #1
253   \socket_use:n{tagsupport/math/end}
254 }
255 \socket_new_plug:nmn
256 {tagsupport/math/display/formula/end}
257 {default}
258 {
259   \socket_use:n{tagsupport/math/substruct/end}
260   \socket_use:n{tagsupport/math/struct/end}
261 }

```

### 5.8.3 Internal sockets

---

#### \l\_\_math\_content\_template\_tl

The default text used as alt or actual text.

```

262 \tl_new:N\l__math_content_template_tl
263 \tl_set:Nn \l__math_content_template_tl
264 {
265   LaTeX~ formula~ starts~
266   \exp_not:N\begin{\g__math_grabbed_env_tl}
267   \c_space_tl
268   \exp_not:V\g__math_grabbed_math_tl
269   \c_space_tl
270   \exp_not:N\end{\g__math_grabbed_env_tl}
271   \c_space_tl LaTeX~ formula~ ends~
272 }

```



---

---

## \l\_\_math\_texsource\_template\_tl

The default text used as texsource

```
273 \tl_new:N\l__math_texsource_template_tl
274 \tl_const:Nn\c__math_inline_env_tl {math}
275 \tl_set:Nn \l__math_texsource_template_tl
276   {
277     \tl_if_eq:NNTF\g__math_grabbed_env_tl\c__math_inline_env_tl
278     {
279       $
280       \exp_not:V\g__math_grabbed_math_tl
281       $
282     }
283     {
284       \exp_not:N\begin{\g__math_grabbed_env_tl}
285       \exp_not:V\g__math_grabbed_math_tl
286       \exp_not:N\end{\g__math_grabbed_env_tl}
287     }
288   }
```

`tagsupport/math/content` (*socket*) The math content is stored in associated files and used for actual and alternative text. As the exact text is still unclear we use a socket to be able to test variants. The socket should set all four tl vars above, if needed to identical values. It can use the two variables `\g__math_grabbed_env_tl` and `\g__math_grabbed_math_tl`

```
289 \socket_new:nn {tagsupport/math/content}{0}
```

Some default sockets to set the contents. TODO: think about naming convention. TODO: think how this should be organized so that one has options to change from the outside and so that there are less repetitions.

`actual+source` (*plug*)

```
290 \socket_new_plug:nnn
291   {tagsupport/math/content}
292   {actual+source}
293   {
294     \tl_set:Ne\l__math_content_actual_tl
295     {
296       \l__math_content_template_tl
297     }
298     \tl_set:Ne \l__math_content_AF_source_tl
299     {
300       \l__math_texsource_template_tl
301     }
302     \tl_set:Nn \l__math_content_AF_mathml_tl {}
303     \tl_set:Nn \l__math_content_alt_tl {}
304   }
```

`alt+source` (*plug*)

```
305 \socket_new_plug:nnn
306   {tagsupport/math/content}
307   {alt+source}
308   {
309     \tl_set:Ne\l__math_content_alt_tl
```

```

310     {
311       \l__math_content_template_tl
312     }
313   \tl_set:Nn \l__math_content_AF_source_tl
314     {
315       \l__math_texsource_template_tl
316     }
317   \tl_set:Nn \l__math_content_AF_mathml_tl {}
318   \tl_set:Nn \l__math_content_actual_tl {}
319 }

320 \socket_assign_plug:nn {tagsupport/math/content}{alt+source}

```

`port/math/struct/begin (socket)` For the main structure we use a socket too. This allow e.g. to create a special one  
`support/math/struct/end (socket)` for luamml which setups additional objects. The begin socket can use the two variables  
`\g__math_grabbed_env_tl` and `\g__math_grabbed_math_tl`

```

321 \socket_new:nn {tagsupport/math/struct/begin}{0}
322 \socket_new:nn {tagsupport/math/struct/end}{0}

```

`default (plug)` TODO: think about some naming convention ...

```

323 \socket_new_plug:nnn
324   {tagsupport/math/struct/begin}
325   {default}
326   {
327     \bool_if:NTF\l__tag_math_texsource_AF_bool
328     { \tl_set_eq:NN \l__math_content_AF_source_tmpa_tl \l__math_content_AF_source_tl }
329     { \tl_clear:N \l__math_content_AF_source_tmpa_tl }
330     \tag_struct_begin:n
331     {
332       tag=Formula,
333       texsource   = \l__math_content_AF_source_tmpa_tl,
334       title-o     = \g__math_grabbed_env_tl,
335       actualtext  = \l__math_content_actual_tl,
336       alt         = \l__math_content_alt_tl
337     }
338   }
339 \socket_new_plug:nnn
340   {tagsupport/math/struct/end}
341   {default}
342   { \tag_struct_end: }
343
344 \socket_assign_plug:nn {tagsupport/math/struct/begin}{default}
345 \socket_assign_plug:nn {tagsupport/math/struct/end}{default}

```

`test-mathml (plug)` This (test-)socket tries to add a mathml-AF to formula. It is activated if a mathml.html has been found and loaded. Additionally it also sets an attribute (this can perhaps be done by default anyway.) As it disturbs the reading of the AF it currently deactivates the /Alt key, unless it has been reenabled with `math/alt/use=true`

```

346 \cs_generate_variant:Nn \str_mdfive_hash:n {o}
347 \tl_new:N\l__math_content_hash_tl

```

we need to save the grabbed math:

```

348 \tl_new:N\l__math_grabbed_math_tl

```

the socket definition

```

349 \socket_new_plug:nnn
350 {tagssupport/math/struct/begin}
351 {test-mathml}
352 {
353   \int_gincr:N\g__math_math_total_int
354   \tl_set:Ne\l__math_content_hash_tl
355   {\str_mdfive_hash:o { \l__math_content_AF_source_tl }}
356   \tl_set_eq:NN\l__math_grabbed_math_tl\g__math_grabbed_math_tl
357   \tl_if_eq:NnTF\g__math_grabbed_env_tl {math}
358   {
359     \tl_set:Nn\l__math_attribute_class_tl{inline}
360   }
361   {
362     \tl_set:Nn\l__math_attribute_class_tl{display}
363   }
364   \bool_if:NF\l__tag_math_alt_bool
365   { \tl_set:Nn \l__math_content_alt_tl{} }

```

debugging option. TODO: hide in debug key.

```

366   \tl_if_exist:cTF { g__math_mathml_ \l__math_content_hash_tl _tl }
367   {
368     \int_gincr:N\g__math_mathml_AF_found_int
369     \bool_if:NTF \l__tag_math_mathml_AF_bool
370     {
371       \int_gincr:N\g__math_mathml_AF_attached_int
372       \typeout {Inserting-mathml~with~Hash~\l__math_content_hash_tl}
373     }
374     {
375       \typeout {Ignoring-mathml~with~Hash~\l__math_content_hash_tl}
376     }
377   }
378   {
379     \typeout{WARNING:-mathml~missing~for~hash~\l__math_content_hash_tl}
380   }
381   \socket_use:n {tagssupport/math/mathml/write} % write hash if request
382   \bool_if:NTF\l__tag_math_texsource_AF_bool
383   { \tl_set_eq:NN \l__math_content_AF_source_tmpa_tl \l__math_content_AF_source_tl }
384   { \tl_clear:N \l__math_content_AF_source_tmpa_tl }
385   \tag_struct_begin:n
386   {
387     tag=Formula,
388     attribute-class=\l__math_attribute_class_tl, %
389     AFref =
390     {
391       \bool_if:NT\l__tag_math_mathml_AF_bool
392       {
393         \cs_if_exist_use:c {g__math_mathml_ \l__math_content_hash_tl _tl}
394       },
395       texsource = \l__math_content_AF_source_tmpa_tl, % should be after mathml AF!
396       title-o = \g__math_grabbed_env_tl, %
397       alt = \l__math_content_alt_tl
398     }
399   }

```

not really needed but looks more symmetric:

```

399 \socket_new_plug:nnn
400   {tagsupport/math/struct/end}
401   {test-mathml}
402   {
403     \tag_struct_end:
404   }

```

`tagsupport/math/substruct/begin (socket)` This holds the code to handle subparts of the formula.

```

tagsupport/math/substruct/end (socket) 405 \socket_new:nn {tagsupport/math/substruct/begin}{0}
406 \socket_new:nn {tagsupport/math/substruct/end}{0}

```

`default (plug)`

```

407 \socket_new_plug:nnn
408   {tagsupport/math/substruct/begin}
409   {default}
410   { \grabaformulapartandstart }
411 \socket_new_plug:nnn
412   {tagsupport/math/substruct/end}
413   {default}
414   {
415     \tagmcend
416     \if@subformulas
417       \tagstructend
418     \fi
419   }
420 \socket_assign_plug:nn {tagsupport/math/substruct/begin}{default}
421 \socket_assign_plug:nn {tagsupport/math/substruct/end}{default}

```

`single (plug)` We need an option to disable subparts as it is unclear if consumers can handle them:

```

422 \socket_new_plug:nnn
423   {tagsupport/math/substruct/begin}
424   {single}
425   {
426     \typeout{====>subpart~splitting~deactivated}
427     \typeout{====>grabbed~math=\meaning\g__math_grabbed_math_tl}
428     \tag_mc_begin:n{ }
429   }
430 \socket_new_plug:nnn
431   {tagsupport/math/substruct/end}
432   {single}
433   { \tag_mc_end: }

```

`tagsupport/math/end (socket)` A socket used at the end of the math (before the closing dollar(s)) which can e.g. set a flag for luamml.

```

434 \socket_new:nn {tagsupport/math/end}{0}

```

`__tag_math_disable:` Similar to the table code we collect the plugs that should be assigned to do nothing if we don't want tagging

```

435 \cs_new_protected:Npn __tag_math_disable:
436   {
437     \socket_assign_plug:nn {tagsupport/math/inline/begin}{noop}
438     \socket_assign_plug:nn {tagsupport/math/inline/end}{noop}
439     \socket_assign_plug:nn {tagsupport/math/inline/formula/begin}{identity}

```

```

440 \socket_assign_plug:nn {tagssupport/math/inline/formula/end}{noop}
441 \socket_assign_plug:nn {tagssupport/math/display/begin}{noop}
442 \socket_assign_plug:nn {tagssupport/math/display/end}{noop}
443 \socket_assign_plug:nn {tagssupport/math/display/formula/begin}{identity}
444 \socket_assign_plug:nn {tagssupport/math/display/formula/end}{noop}
445 }

```

(End of definition for `\_tag_math_disable:`.)

`\_tag_math_enable:` Similar to the table code we collect the default plugs that should be assigned if we want tagging

```

446 \cs_new_protected:Npn \_tag_math_enable:
447 {
448 \socket_assign_plug:nn {tagssupport/math/inline/begin}{MC}
449 \socket_assign_plug:nn {tagssupport/math/inline/end}{MC}
450 \socket_assign_plug:nn {tagssupport/math/inline/formula/begin}{default}
451 \socket_assign_plug:nn {tagssupport/math/inline/formula/end}{default}
452 \socket_assign_plug:nn {tagssupport/math/display/begin}{default}
453 \socket_assign_plug:nn {tagssupport/math/display/end}{default}
454 \socket_assign_plug:nn {tagssupport/math/display/formula/begin}{default}
455 \socket_assign_plug:nn {tagssupport/math/display/formula/end}{default}
456 }

```

(End of definition for `\_tag_math_enable:`.)

At begin document we can activate:

```

457 \AtBeginDocument{\tag_if_active:T{\_tag_math_enable: }}

```

## 5.9 Interface commands

`\_math_process:nn` A no-op place-holder; the internal wrapper means that it does not need to be concerned with internals.

```

\_math_process:Vn
\_math_process_auxi:nn
\_math_process_auxii:nn
458 \cs_new_protected:Npn \_math_process:nn #1#2
459 {
460 \legacy_if:nF { measuring@ }
461 {
462 \tl_if_in:nnTF {#2} { \m@th }
463 { \bool_set_true:N\l__math_fakemath_bool }
464 { \tl_trim_spaces_apply:nN {#2} \_math_process_auxi:nn {#1} }
465 }
466 }
467 \cs_generate_variant:Nn \_math_process:nn { V }
468 \cs_new_protected:Npn \_math_process_auxi:nn #1#2
469 {
470 \tl_gset:Nn \g__math_grabbed_env_tl {#2}
471 \tl_gset:Nn \g__math_grabbed_math_tl {#1}
472 \_math_process_auxii:nn {#2} {#1}
473 }
474 \cs_new_protected:Npn \_math_process_auxii:nn #1#2 { }

```

(End of definition for `\_math_process:nn`, `\_math_process_auxi:nn`, and `\_math_process_auxii:nn`.)

`\math_processor:n` A simple installer

```

475 \cs_new_protected:Npn \math_processor:n #1
476 { \cs_set_protected:Npn \_math_process_auxii:nn ##1##2 {#1} }

```

(End of definition for `\math_processor:n`. This function is documented on page 3.)

## 5.10 Content grabbing

Grab up to a single \$, for inline math mode, suppressing any processing if the token is `\m@th` found in the content.

what's that test doing?

It is some kind of fix, to avoid the remote possibility that the math is empty, making the code produce an unwanted `$$`.

cf. the code for this in `\@ensuredmath`

It is harmless but unnecessary in the `dollardollar` grabbing below.

what's that test doing?

```

477 \cs_new_protected:Npn \__math_grab_dollar:w % $
478   #1 $
479   {
480     \tl_if_blank:nF {#1}
481     {
482       \__math_process:nn { math } {#1} % $
483       \bool_lazy_or:nTF
484         {\legacy_if_p:n { measuring@ }}
485         { \l__math_fakemath_bool }
486         { #1 $ }
487         {
488           \socket_use:n {tagsupport/math/inline/begin} %end P-MC
489           \socket_use:nn{tagsupport/math/inline/formula/begin}{#1}
490           $
491           \socket_use:n {tagsupport/math/inline/formula/end}
492           \socket_use:n {tagsupport/math/inline/end} % restart P-MC
493         }
494     }
495 }

```

(End of definition for `\__math_grab_dollar:w`.)

And for the classical  $\TeX$  display structure.

```

496 \cs_new_protected:Npn \__math_grab_dollardollar:w % $$
497   #1 $$
498   {
499     \tl_if_blank:nF {#1}
500     {
501       \__math_process:nn { equation* } {#1}
502       \socket_use:n {tagsupport/math/display/begin}
503       \socket_use:nn{tagsupport/math/display/formula/begin}{#1}
504       $$
505     }
506 }

```

The end code is added through a `\aftergroup` so we store it inside a command.

```

507 \cs_new_protected:Npn \__math_tag_dollardollar_display_end:
508   {
509     % \typeout{== tag dollarldollar display end}
510     % \ShowTagging{struct-stack}
511     \para_raw_end:
512     \tagpdfpara0n
513     \l__math_tmpa_skip \lastskip
514     \socket_use:n{tagsupport/math/display/formula/end}
515     \penalty \postdisplaypenalty

```

TODO why is that needed? where is para-tagging disabled?

This reinserts the below display skips. It must be doubled to get the right amount:

```

516 \skip_vertical:n { -\l__math_tmpa_skip * 2 }
517 %
518 \@doendpe           % this has no \end{...} to take care of it
519 }
520

```

(End of definition for `\__math_grab_dollardollar:w`.)

`\__math_grab_inline:w` Collect inline math content and deal with the need to move to math mode.

```

521 \cs_new_protected:Npn \__math_grab_inline:w % \{
522   #1 \)
523   {
524     \tl_if_blank:nF {#1}
525     {
526       $ #1 $
527     }
528     \bool_set_false:N \l__math_collected_bool
529   }

```

(End of definition for `\__math_grab_inline:w`.)

`\__math_grab_eqn:w` For the most common use of `\[/\]`: turn into an environment.

```

530 \cs_new_protected:Npn \__math_grab_eqn:w % \[
531   #1 \]
532   {
533     % \typeout{collected? = \bool_if:NTF \l__math_collected_bool {true}{false}}
534     \begin { equation* } #1 \end { equation* }
535   }

```

(End of definition for `\__math_grab_eqn:w`.)

## 5.11 Marking math environments

A general mechanism for math mode environments that do not grab their content (*cf.* most `amsmath` environments).

---

`\l__math_env_name_tl` To allow us to carry out “special effects”

```

536 \tl_new:N \l__math_env_name_tl

```

Here we set up specialised handling of environments. The idea for the `arg-spec` key is that if an environment takes arguments, we don’t worry during the main grabbing. Rather, we remove the arguments from the grabbed content and forward only the payload. That is done by (ab)using `lcmd`.

```

537 \keys_define:nn { __math }
538   {
539     arg-spec .code:n =
540       {
541         \ExpandArgs { c } \DeclareDocumentCommand
542           { __math_env \l__math_env_name_tl _aux: }
543           {#1}
544           { \__math_env_forward:w }
545       }
546   }

```

$\backslash\text{math\_register\_env:nn}$   
 $\backslash\text{math\_register\_env:n}$   
 $\backslash\text{RegisterMathEnvironment}$

Set up to capture environment content and make available.

```

547 \cs_new_protected:Npn \math_register_env:nn #1#2
548 {
549   \tl_set:Nn \l__math_env_name_tl {#1}
550   \keys_set:nn { __math } {#2}
551   \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
552   \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
553   %
554   \ExpandArgs { nne } \RenewDocumentEnvironment {#1} { b }
555   {
556     \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
557     {
558       \typeout{===>B1}
559     }
560     {
561       \typeout{===>B2}
562       \cs_if_exist:cTF { __math_env_ #1 _aux: }
563       {
564         \exp_not:c { __math_env_ #1 _aux: }
565         ##1 \exp_not:N \__math_env_end: {#1}
566       }
567       { \exp_not:N \__math_process:nn {#1} {##1} }
568       \exp_not:n { \@kernel@math@registered@begin }
569       \bool_set_true:N \exp_not:N \l__math_collected_bool
570     }
571     \exp_not:N \tracingall
572     \exp_not:c { __math_env_ #1 _begin: }
573     ##1
574     \exp_not:c { __math_env_ #1 _end: }
575     \exp_not:N \tracingnone
576   }
577   {
578   }
579 }
580
581 \cs_new_protected:Npn \math_register_halign_env:nn #1#2
582 {
583   \tl_set:Nn \l__math_env_name_tl {#1}
584   \keys_set:nn { __math } {#2}
585   \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
586   \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
587   %
588   \ExpandArgs { nnee } \RenewDocumentEnvironment {#1} { b }
589   {
590     \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
591     {
592       \typeout{===>B1}
593     }
594     {
595       \typeout{===>B2}
596       \cs_if_exist:cTF { __math_env_ #1 _aux: }
597       {
598         \exp_not:c { __math_env_ #1 _aux: }
599         ##1 \exp_not:N \__math_env_end: {#1}

```



```

600     }
601     { \exp_not:N \l__math_process:nn {#1} {##1} }
602     \exp_not:n { \@kernel@math@registered@begin }
603     \bool_set_true:N \exp_not:N \l__math_collected_bool
604   }
605 %   \exp_not:N \tracingall
606   \exp_not:c { __math_env_ #1 _begin: }
607   ##1
608 %   \exp_not:N \tracingnone
609 }
610 {
611   \exp_not:c { __math_env_ #1 _end: }
612 }
613 }

```

TODO: the following command is neither documented nor used. Is it needed?

```

614 \cs_new_protected:Npn \math_register_odd_env:nn #1#2
615 {
616   \tl_set:Nn \l__math_env_name_tl {#1}
617   \keys_set:nn { __math } {#2}
618   \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
619   \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
620 %
621   \ExpandArgs { nnee } \RenewDocumentEnvironment {#1} { b }
622   {
623     \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
624     {
625 %       \typeout{===>B1}
626     }
627     {
628 %       \typeout{===>B2}
629       \cs_if_exist:cTF { __math_env_ #1 _aux: }
630       {
631         \exp_not:c { __math_env_ #1 _aux: }
632         ##1 \exp_not:N \l__math_env_end: {#1}
633       }
634       { \exp_not:N \l__math_process:nn {#1} {##1} }
635       \exp_not:n { \@kernel@math@registered@begin }
636       \bool_set_true:N \exp_not:N \l__math_collected_bool
637     }
638 %   \exp_not:N \tracingall
639   \exp_not:c { __math_env_ #1 _begin: }
640   ##1
641 }
642 {
643   \exp_not:c { __math_env_ #1 _end: }
644 % needed if we don't have $$...$$
645 %   \exp_not:n { \typeout{---> @kernel@math@registered@end } }
646   \exp_not:n { \@kernel@math@registered@end }
647 }
648 }
649
650
651 % FMi: compare with block change!

```

```

652 %
653 % \DeclareRobustCommand*\begin[1]{%
654 % \UseHook{env/#1/before}%
655 % \@ifundefined{#1}%
656 % {\def\reserved@a{\@latex@error{Environment #1 undefined}\@eha}}%
657 % {\def\reserved@a{\def\@currenvir{#1}%
658 % \edef\@currenvline{\on@line}%
659 % \execute@begin@hook{#1}%
660 % \csname #1\endcsname}}%
661 % \@ignorefalse
662 % \begingroup
663 % \@endpefalse % tmp!!! is it ok to drop this here?
664 % \reserved@a}
665
666
667 \cs_new:Npn \@kernel@math@registered@begin {
668 % \ShowTagging{struct-stack}
669 %\typeout{==>A1}\ShowTagging{struct-stack,mc-current}
670 \mode_if_vertical:TF
671 {
672 % \legacy_if:nTF { @endpe }
673 % { \legacy_if_set_false:n { @endpe } }
674 % { \__block_list_beginpar_vmode: }
675 %
676 % \typeout{==>~ at:~ \g__tag_struct_tag_tl}
677 %
678 \tag_if_active:T
679 {
680 \exp_args:Noo\str_if_eq:nmF \g__tag_struct_tag_tl { \l__tag_para_main_tag_tl }
681 {
682 % \typeout{==>A2}
683 \__block_beginpar_vmode:
684 } % needs correction!
685 }
686 }
687 {
688 % \typeout{==>A3}
689 \__tag_tool_close_P:
690 }
691 \socket_use:nn{tagsupport/math/display/formula/begin}{}
692 \tagpdfparaOff
693 % \typeout{==>MC1}\ShowTagging{mc-current}
694 }
695
696 \cs_new:Npn \@kernel@math@registered@end {
697 % \typeout{==>MC2}\ShowTagging{mc-current}
698 \para_raw_end:
699 \tagpdfparaOn
700 \socket_use:n{tagsupport/math/display/formula/end}
701 % \typeout{==>MC3}\ShowTagging{mc-current}
702 \@endpetrue
703 }
704
705 \cs_new_protected:Npn \math_register_env:n #1

```

```

706 { \math_register_env:nn {#1} { } }
707
708 \NewDocumentCommand \RegisterMathEnvironment { 0{ } m }
709 { \math_register_env:nn {#2} {#1} }

```

(End of definition for `\math_register_env:nn`, `\math_register_env:n`, and `\RegisterMathEnvironment`. These functions are documented on page 3.)

`\_math_env_forward:w`

```

710 \cs_new_protected:Npn \_math_env_forward:w #1 \_math_env_end: #2
711 { \_math_process:nn {#2} {#1} }

```

(End of definition for `\_math_env_forward:w`.)

## 5.12 Document commands

Add one more here: `displaymath`, which is equivalent to `\[ , \]` and hence to the basic `equation*`. Added in more recent branch.

```

\equation
\_math_equation_begin:
\equation*
\_math_equation_star_begin:
\endequation
\_math_equation_end:
\endequation*
\_math_equation_star_end:

```

These environments are not set up by `amsmath` to collect their body, so we do that here. This has to be done *after* we can be sure `amsmath` is loaded.

Note that with `amsmath` loaded, `equation*` and `equation` are the two basics: they are used to define the other single-row display environments, etc.

```

712 \tl_gput_right:Nn \@kernel@before@begindocument
713 {
714   \math_register_env:n { equation }
715   \math_register_env:n { equation* }
716   % at the moment register_env can only do display math
717   %   \math_register_env:n { math }
718   \RenewDocumentEnvironment{math} {b}{\$#1$}{ }
719   % and this one doesn't work either
720   %   \math_register_env:n { displaymath }
721   \RenewDocumentEnvironment{displaymath} {b}{\[#1\]}{ }
722 }

```

(End of definition for `\equation` and others. These functions are documented on page ??.)

`\(` If math mode has not been collected, we need to do that; otherwise, worry about whether  
`\)` we are in math mode or not. The closing command here can only occur inside a collected math block: otherwise it will be simply used as a delimiter.

```

723 \cs_gset_protected:Npn \(\ % \)
724 {
725   \bool_if:NTF \l_math_collected_bool
726   {
727     \mode_if_math:TF
728     { \badmath }
729     { $ }
730   }
731   {
732     \_math_grab_inline:w
733   }

```

```

734 } % \langle
735 \cs_gset_protected:Npn \langle
736 {
737   \mode_if_math:TF
738     { $ }
739     { \@badmath }
740 }

```

(End of definition for  $\langle$  and  $\rangle$ . These functions are documented on page ??.)

$\lceil$  Again, we need to watch for when amsmath is loaded after this code. The flag usage here  
 $\rfloor$  is to cover the case where  $\lceil/\rfloor$  is hidden inside another environment. In this case the grabbing happens on the outer level and should not be repeated.

```

741 \tl_gput_right:Nn \@kernel@before@begindocument
742 {
743   \cs_gset_protected:Npn \lceil % \rfloor
744     {
745       \__math_grab_eqn:w
746       % \bool_if:NTF \l__math_collected_bool
747       %   { \begin { equation* } }
748       %   { \__math_grab_eqn:w }
749     } % \lceil
750   \cs_gset_protected:Npn \rfloor
751     {
752       \@badmath
753       % \bool_if:NTF \l__math_collected_bool
754       %   { \end{ equation* } }
755       %   { \@badmath }
756     }
757 }

```

(End of definition for  $\lceil$  and  $\rfloor$ . These functions are documented on page ??.)

why does ensuremath need handling at all?

Indeed! Currently, this is setup to process the math that it has anyways already captured as its argument; thus it is more efficient than leaving the capture to be repeated by the `\everymath`

A bit of nesting fun to make sure we collect only if required.

```

758 %\cs_gset_protected:Npn \ensuremath #1
759 % {
760 %   \mode_if_math:TF
761 %     {#1}
762 %     {
763 %       \bool_if:NTF \l__math_collected_bool
764 %         { \@ensuredmath {#1} }
765 %         {
766 %           \bool_set_true:N \l__math_collected_bool
767 %           \__math_process:nn { math } {#1}
768 %           \@ensuredmath {#1}
769 %           \bool_set_false:N \l__math_collected_bool
770 %         }
771 %     }
772 % }

```

(End of definition for `\ensuremath`. This function is documented on page ??.)

### 5.13 `\everymath` and `\everydisplay`

The business end for grabbing inline math and “raw” T<sub>E</sub>X display. Most display math mode is actually handled elsewhere, as we have macro control.

```
773
774 \exp_args:No \tex_everymath:D
775 {
776   \tex_the:D \tex_everymath:D
777   \bool_if:NF \l__math_collected_bool
778   {
779     \bool_set_true:N \l__math_collected_bool
780     \__math_grab_dollar:w
781   }
782 }
783
784 \exp_args:No \tex_everydisplay:D
785 {
786   \tex_the:D \tex_everydisplay:D
787   \iftrue % this may have to be a settable flag!
788   % \typeout{==>~ in~ everydisplay}
```

flipping the `\belowdisplay` values is done so that we get (assumption) a negative skip and not make the page bigger then we take that out, then we add the tagging code (in `\__math_tag_dollardollar_display_end`) and then we put a real `\postdisplaypenalty` in and the right skip (of which we don't know if it is short or a normal `\belowdisplayskip`). This might need some refinement if that skip is actually negative from the start (not sure it ever is and is worth bothering about)

```
789     \skip_set:Nn \belowdisplayskip {-\belowdisplayskip}
790     \skip_set:Nn \belowdisplayshortskip {-\belowdisplayshortskip}
791     \int_set:Nn \postdisplaypenalty {10000}
792     \group_insert_after:N \__math_tag_dollardollar_display_end:
793     \fi
794     \bool_if:NF \l__math_collected_bool
795     {
796       \bool_set_true:N \l__math_collected_bool
797       \__math_grab_dollardollar:w
798     }
799 }
```

### 5.14 Modifying kernel environments

We need to cover this even though it is, of course, not encouraged.

```
800 \math_register_env:n { eqnarray }
801 \math_register_env:n { eqnarray* }
      Tabulars currently contain a $ that shouldn't trigger math tagging.
802 \RequirePackage{array}
803 \tl_if_in:NnT\@tabular{$}
804 {
805   \def\@tabular{%
806     \leavevmode
807     \UseTaggingSocket{tbl/hmode/begin}%
808     \hbox \bgroup
809     \bool_set_true:N \l__math_collected_bool
```

```

810 $
811 \bool_set_false:N \l__math_collected_bool
812 \col@sep\tabcolsep \let\d@llarbegin\begin\group
813                                     \let\d@llarend\endgroup
814 \@tabarray}
815 }

```

`\__math_m@th:` Handle non-math use of math mode. At present nesting isn't supported as `\m@th` pops up in a few places that *are* math mode!

```

816 \cs_new_eq:NN \__math_m@th: \m@th
817 \cs_gset_protected:Npn \m@th
818 {
819   \bool_set_true:N \l__math_collected_bool
820   \__math_m@th:
821 }

```

(End of definition for `\__math_m@th:` and `\m@th`. This function is documented on page ??.)

## 5.15 Disable math grabbing in the begindocument hook

For example `amsart` uses `math` to measure text there.

```

822 \tl_gput_right:Nn \@kernel@before@begindocument
823 {
824   \bool_set_true:N \l__math_collected_bool
825 }
826 \tl_gput_right:Nn \@kernel@after@begindocument
827 {
828   \bool_set_false:N \l__math_collected_bool
829 }

```

## 5.16 Modifying `amsmath`

`\__math_amsmath_align@:n` Mark up all of the display environments as the content is captured anyway. We then use an internal macro in each environment type to insert the processing code. Each of these is slightly different, so we cannot use a simple loop here. The test for `\split@tag` is required as the `split` environment internally uses `gather` *when not within an `amsmath` environment*, for example inside `equation`. Without the precaution, we'd get two copies of the grabbed math, the second of which would start with `\split@tag`.

```

830
831
832
833 \tl_gput_right:Nn \@kernel@before@begindocument {
834 %
835 \renewenvironment{gather*}{%
836   \start@gather\st@rredtrue
837 }
838 {%
839 % this redirection doesn't work if we alter "gather"!
840 % \endgather
841 % so replace it with its real meaning
842 \math@cr \black@\totwidth@ \egroup
843 $$\ignorespacesafterend
844 }

```

```

845 \def\common@align@ending {
846   \math@cr \black@\totwidth@
847   \egroup
848   \ifingather@
849     \restorealignstate@
850     \egroup
851     \nonumber
852     \ifnum0='{ \fi\iffalse}\fi
853   \else
854     $$%
855   \fi
856   \ignorespacesafterend
857 }
858 \renewenvironment{alignat}{%
859   \start@align\z@\st@rredfalse
860 }{%
861   \common@align@ending
862 }
863 \renewenvironment{alignat*}{%
864   \start@align\z@\st@rredtrue
865 }{%
866   \common@align@ending
867 }
868 \renewenvironment{xalignat}{%
869   \start@align\@ne\st@rredfalse
870 }{%
871   \common@align@ending
872 }
873 \renewenvironment{xalignat*}{%
874   \start@align\@ne\st@rredtrue
875 }{%
876   \common@align@ending
877 }
878 \renewenvironment{xxalignat}{%
879   \start@align\tw@\st@rredtrue
880 }{%
881   \common@align@ending
882 }
883 \renewenvironment{align}{%
884   \start@align\@ne\st@rredfalse\m@ne
885 }{%
886   \common@align@ending
887 }
888 \renewenvironment{align*}{%
889   \start@align\@ne\st@rredtrue\m@ne
890 }{%
891   \common@align@ending
892 }
893 \renewenvironment{flalign}{%
894   \start@align\tw@\st@rredfalse\m@ne
895 }{%
896   \common@align@ending
897 }
898 \renewenvironment{flalign*}{%

```

```

899 \start@align\tw@\st@rredtrue\m@ne
900 }{%
901 \common@align@ending
902 }
903 %
904 \renewenvironment{multline*}{\start@multline\st@rredtrue}
905 {%
906 \iftagsleft@ \exp\lendmultline@ \else \exp\rendmultline@ \fi
907 \ignorespacesafterend
908 }

```

Also for false?

```

909 \def\measuring@true{\let\ifmeasuring@\iftrue\tag_stop:}
910 %
911 \math_register_halign_env:nn {align}{}
912 \math_register_halign_env:nn {align*}{}
913 \math_register_halign_env:nn {alignat}{}
914 \math_register_halign_env:nn {alignat*}{}
915 \math_register_halign_env:nn {flalign}{}
916 \math_register_halign_env:nn {flalign*}{}
917 \math_register_halign_env:nn {gather}{}
918 \math_register_halign_env:nn {gather*}{}
919 \math_register_halign_env:nn {multline}{}
920 \math_register_halign_env:nn {multline*}{}
921 \math_register_halign_env:nn {xalignat}{}
922 \math_register_halign_env:nn {xalignat*}{}
923 \math_register_halign_env:nn {xxalignat}{}
924 %
925 \@namedef{maketag @ @ @} #1{%
926 % \typeout{--->maketag @ @ @}
927 \ifmeasuring@
928 \hbox{\m@th\normalfont#1}%
929 \else
930 \tagmccend \tagstructbegin{tag=Lbl}%
931 \tagmccbegin{tag=Lbl}%
932 \hbox{\m@th\normalfont#1}%
933 \tagmccend \tagstructend \tagmccbegin{}%
934 \fi
935 }
936 \@namedef{math@cr @ @ @ gather} {%
937 \ifst@rred\nonumber\fi
938 &\relax
939 \make@display@tag
940 %
941 \maybestartnewformulatag
942 %
943 \ifst@rred\else\global\@eqnswtrue\fi
944 \global\advance\row@\@ne
945 \cr
946 }
947 \@namedef{math@cr @ @ @ align} {%
948 \ifst@rred\nonumber\fi
949 \if@eqnsw \global\tag@true \fi

```



```

950 \global\advance\row@\@ne
951 \add@amps\maxfields@
952 \omit
953 \kern-\alignsep@
954 \iftag@
955   \setboxz@h{\@lign\strut@{\make@display@tag}}%
956   \place@tag
957 \fi
958 %
959   \maybestartnewformulatag
960 %
961 \ifst@rred\else\global\@eqnswtrue\fi
962 \global\lineht@z@
963 \cr
964 }
965 \def\restore@math@cr{\@namedef{math@cr @ @ @}{
966 %
967   \maybestartnewformulatag
968 %
969   \cr}}
970 \restore@math@cr
971 }

```

(End of definition for `\_math_amsmath_align@:nn` and others. These functions are documented on page ??.)

`\_math_split_at_nl:NN` This splits grabbed math at newlines.

```

972 \cs_new:Npn \_math_split_at_nl:NN #1#2 {
973   \tl_set:Nf \l_math_tmpa_tl {
974     \exp_after:wN \_math_split_at_nl_first:w #1 \ \q_nil \ \s_stop }
975   \exp_after:wN \_math_split_at_nl_aux:nnNN \l_math_tmpa_tl #1 #2
976 }

```

and the auxiliary commands

```

977 \cs_new:Npn \_math_split_at_nl_first:w #1 \ \ #2 \ \ #3 \s_stop
978 {
979   \quark_if_nil:nTF {#2}
980   { {#1} { } }
981   {
982     \_math_split_chk_if_begin:ww #1 \begin \q_nil \s_mark
983     #2 \ \ #3 \s_stop
984   }
985 }
986
987 \cs_new_protected:Npn \_math_split_at_nl_aux:nnNN #1 #2 #3 #4
988 {
989   \tl_gset:Nn #4 {#1}
990   \tl_gset:Nn #3 {#2}
991 }
992
993 \cs_new:Npn \_math_split_chk_if_begin:ww
994 #1 \begin #2 #3 \s_mark #4 \ \ \q_nil \ \ \s_stop
995 {
996   \quark_if_nil:nTF {#2}

```

```

997     { {#1} {#4} }
998     {
999         \exp_after:wN \_math_split_collect_one_end:w
1000         \_math_split_cleanup_begin_q_nil:w #1 \begin{#2} #3 \ \ #4 \s_stop
1001         { } { 1 }
1002     }
1003 }
1004
1005 \cs_new:Npn \_math_split_cleanup_begin_q_nil:w #1 \begin \q_nil {#1}
1006
1007 \cs_new:Npn \_math_split_collect_one_end:w #1 \end #2 #3 \s_stop #4 #5
1008 {
1009     \exp_args:Nf \_math_split_check_count_begins:n
1010     { \_math_split_count_begins:n { #4 #1 } } {#5}
1011     { #4 #1 \end{#2} } {#3}
1012 }
1013 \cs_new:Npn \_math_split_count_begins:n #1
1014 { \int_eval:n { 0 \_math_split_count_begins:w #1 \begin \q_nil } }
1015
1016 \cs_new:Npn \_math_split_count_begins:w #1 \begin #2
1017 { \quark_if_nil:nF {#2} { +1 \_math_split_count_begins:w } }
1018
1019 \cs_new:Npn \_math_split_check_count_begins:n #1 #2 #3 #4
1020 {
1021     \int_compare:nNnTF {#1} = {#2}
1022     {
1023         \exp_last_unbraced:Nf \_math_split_final_cleanup:nn
1024         { \_math_split:n { \_math_split_guard:n {#3} #4 } }
1025     }
1026     {
1027         \exp_args:No \use_ii_i:nn
1028         { \exp_after:wN { \int_value:w \int_eval:n { #2 + 1 } } }
1029         { \_math_split_collect_one_end:w #4 \s_stop {#3} }
1030     }
1031 }
1032 \cs_new:Npn \_math_split_final_cleanup:nn #1 #2
1033 {
1034     \exp:w \_math_split_final_cleanup:w #1
1035     \_math_split_guard:n \q_nil \s_mark { }
1036     {#2}
1037 }
1038 \cs_new:Npn \_math_split_final_cleanup:w #1 \_math_split_guard:n #2 #3 \s_mark #4
1039 {
1040     \quark_if_nil:nTF {#2}
1041     { \exp_end: { #4 #1 } }
1042     { \_math_split_final_cleanup:w #3 \s_mark { #4 #1 #2 } }
1043 }
1044
1045 \cs_new:Npn \_math_split:n #1 {
1046     \_math_split_at_nl_first:w #1 \ \ \q_nil \ \ \s_stop }
1047
1048 % this looks unused.
1049 %\NewDocumentCommand \splitnl { mm +m }
1050 % {

```

```

1051 % \tl_set:Nf \l__math_tmpa_tl { \split:n {#3} }
1052 % \show \l__math_tmpa_tl
1053 % \exp_after:wN \__splitnl_aux:nnNN \l__math_tmpa_tl #1 #2
1054 % }

```

(End of definition for `\__math_split_at_nl:NN`.)

`\maybestartnewformulatag`

```

1055
1056 \newif\if@subformulas
1057 \tl_new:N \result
1058
1059 \cs_new_protected:Npn\grabaformulapartandstart {
1060   \__math_split_at_nl:NN \g__math_grabbed_math_tl \result
1061   \typeout{====>first-result=\meaning\result}
1062   \typeout{====>first-tmpmathcontent=\meaning\g__math_grabbed_math_tl}
1063   \tl_if_empty:NTF \g__math_grabbed_math_tl
1064   {
1065     \typeout{====>formula~ has~ no~ subparts}
1066     \global\@subformulasfalse
1067   }
1068   {
1069     \typeout{====>formula~ has~ subparts}
1070     \global\@subformulastrue
1071     \edef\resulttitle{\g__math_grabbed_env_tl\space (part)}
1072     \tagstructbegin{tag=Formula,

```

For now we don't put real content in `/alt` or `/ActualText` on subformulas but we add a short text to satisfy the pdf/ua-2 validator

```

1073 %       alt=\result,
1074 %       alt = subformula,
1075 %       title-o=\resulttitle
1076 %     }
1077 %   }
1078 %   \tagmcbegin{}
1079 % }
1080
1081 \cs_new_protected:Npn\grabaformulapartandmayberestart {
1082   \__math_split_at_nl:NN \g__math_grabbed_math_tl \result
1083   \typeout{====>result=\meaning\result}
1084   \typeout{====>tmpmathcontent=\meaning\g__math_grabbed_math_tl}
1085 % \tl_if_empty:NTF \g__math_grabbed_math_tl
1086 % {
1087 %   \typeout{====>tmpmathcontent=empty}
1088 % }
1089 % {
1090 %   \typeout{====>tmpmathcontent=not-empty}
1091 %   \edef\resulttitle{\g__math_grabbed_env_tl\space (part)}
1092 %   \tagstructbegin{tag=Formula,
1093 %     alt=\result,
1094 %     title-o=\resulttitle
1095 %   }
1096 % }
1097 %   \tagmcbegin{}
1098 % }

```

(End of definition for `\maybestartnewformulatag`. This function is documented on page ??.)

```

1099 \def\maybestartnewformulatag {
1100 \if@subformulas
1101 \ifmeasuring@\else
1102 %
1103 \tl_if_empty:NF \g__math_grabbed_math_tl
1104 {
1105 \tagmcentd
1106 \tagstructend
1107 \grabaformulapartandmayberestart
1108 }
1109 \fi
1110 \fi
1111 }

```

The `breqn` packages changes catcodes and that isn't yet covered by our mechanism.

```

1112 %\AddToHook{package/breqn/after}{
1113 % \typeout{===>~ in~ hook}
1114 % \math_register_halign_env:nn {dmath}{}
1115 % \math_register_halign_env:nn {dgroup*}{}
1116 %}
1117 \ExplSyntaxOff
1118 <@@=
1119 </kernel>

```

## Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Symbols	
<code>\#</code> .....	68, 137
<code>\%</code> .....	69
<code>\(</code> .....	521
<code>\(</code> .....	<u>723</u>
<code>\)</code> .....	522
<code>\)</code> .....	<u>723</u>
<code>@@</code> commands:	
<code>\l_@@_collected_bool</code> .....	3, 7
<code>\l_@@_fakemath_bool</code> .....	7
<code>\[</code> .....	530, <u>721</u>
<code>\[</code> .....	8, <u>23</u> , <u>27</u> , <u>28</u> , <u>741</u>
<code>\</code> .....	974, 977, 983, 994, 1000, 1046
<code>\{</code> .....	66
<code>\}</code> .....	67
<code>\]</code> .....	531, <u>721</u>
<code>\]</code> .....	8, <u>23</u> , <u>27</u> , <u>28</u> , <u>741</u>
<b>A</b>	
actual+source (plug) .....	<u>290</u>
<code>\AddToHook</code> .....	7, 9, 119, 159, 176, 1112
<code>\advance</code> .....	944, 950
<code>\aftergroup</code> .....	22
alt+source (plug) .....	305
<code>\AssignSocketPlug</code> .....	118, 155, 156, 157, 158
<code>\AtBeginDocument</code> .....	457
<b>B</b>	
<code>\begin</code> .....	266, 284, 534, 653, 747, 982, 994, 1000, 1005, 1014, 1016
<code>\begingroup</code> .....	65, 662, 812
<code>\belowdisplay</code> .....	29
<code>\belowdisplayshortskip</code> .....	790
<code>\belowdisplayskip</code> .....	29, 789
<code>\bgroup</code> .....	808
block internal commands:	
<code>\_block_beginpar_vmode:</code> .....	683
<code>\_block_list_beginpar_vmode:</code> ..	674
bool commands:	
<code>\bool_gset_true:N</code> .....	104, 145

<code>\bool_if:NTF</code> . . . . .	58, 152, 327, 364, 369, 382, 390, 533, 556, 590, 623, 725, 746, 753, 763, 777, 794	<code>default (plug)</code> . . . . .	<a href="#">214</a> , <a href="#">236</a> , <a href="#">245</a> , <a href="#">323</a> , <a href="#">407</a>
<code>\bool_lazy_or:mnTF</code> . . . . .	483	<code>\displaylines</code> . . . . .	8
<code>\bool_new:N</code> 10, 11, 36, 37, 38, 39, 40, 41		<b>E</b>	
<code>\bool_set_false:N</code> . . . . .	528, 769, 811, 828	<code>\edef</code> . . . . .	658, 1071, 1091
<code>\bool_set_true:N</code> . . . . .	463, 569, 603, 636, 766, 779, 796, 809, 819, 824	<code>\egroup</code> . . . . .	842, 847, 850
bool internal commands:		<code>\else</code> . . . . .	853, 906, 929, 943, 961, 1101
<code>\l_math_collected_bool</code> 9, 10, 528, 533, 556, 569, 590, 603, 623, 636, 725, 746, 753, 763, 766, 769, 777, 779, 794, 796, 809, 811, 819, 824, 828		<code>\end</code> . . . . .	270, 286, 518, 534, 754, 1007, 1011
<code>\l_math_fakemath_bool</code> 9, 11, 463, 485		end internal commands:	
box commands:		<code>\_math_tag_dollardollar_-</code> <code>display_end</code> . . . . .	29
<code>\box_new:N</code> . . . . .	128	<code>\endcsname</code> . . . . .	660
box internal commands:		<code>\endequation</code> . . . . .	712
<code>\l_math_tmpa_box</code> . . . . .	128, 131	<code>\endequation*</code> . . . . .	712
<code>\boxed</code> . . . . .	7	<code>\endgather</code> . . . . .	840
<b>C</b>			
char commands:		<code>\endgroup</code> . . . . .	73, 813
<code>\char_set_catcode_other:N</code> . . . . .	66, 67, 68, 69, 137	<code>\ensuremath</code> . . . . .	7, 758
clist commands:		<code>\equalign</code> . . . . .	8
<code>\clist_map_inline:Nn</code> . . . . .	139	<code>\equation</code> . . . . .	712
<code>\clist_new:N</code> . . . . .	47	<code>\equation*</code> . . . . .	712
<code>\clist_put_right:Nn</code> . . . . .	48	<code>\everydisplay</code> . . . . .	29
<code>\cr</code> . . . . .	945, 963, 969	<code>\everymath</code> . . . . .	3, 28, 29
cs commands:		exp commands:	
<code>\cs_generate_variant:Nn</code> 189, 346, 467		<code>\exp:w</code> . . . . .	1034
<code>\cs_gset_eq:NN</code> . . . . .	551, 552, 585, 586, 618, 619	<code>\exp_after:wN</code> 974, 975, 999, 1028, 1053	
<code>\cs_gset_protected:Npn</code> . . . . .	723, 735, 743, 750, 758, 817	<code>\exp_args:Nf</code> . . . . .	1009
<code>\cs_if_exist:NTF</code> . . . . .	562, 596, 629	<code>\exp_args:No</code> . . . . .	774, 784, 1027
<code>\cs_if_exist_use:N</code> . . . . .	392	<code>\exp_args:Noo</code> . . . . .	680
<code>\cs_new:Npn</code> . . . . .	667, 696, 972, 977, 993, 1005, 1007, 1013, 1016, 1019, 1032, 1038, 1045	<code>\exp_end:</code> . . . . .	1041
<code>\cs_new_eq:NN</code> . . . . .	816	<code>\exp_last_unbraced:Nf</code> . . . . .	1023
<code>\cs_new_protected:Npn</code> . . . . .	21, 49, 64, 72, 129, 180, 435, 446, 458, 468, 474, 475, 477, 496, 507, 521, 530, 547, 581, 614, 705, 710, 987, 1059, 1081	<code>\exp_not:N</code> . . . . .	266, 270, 284, 286, 556, 564, 565, 567, 569, 571, 572, 574, 575, 590, 598, 599, 601, 603, 605, 606, 608, 611, 623, 631, 632, 634, 636, 638, 639, 643
<code>\cs_set_eq:NN</code> . . . . .	138	<code>\exp_not:n</code> . . . . .	268, 280, 285, 568, 602, 635, 645, 646
<code>\cs_set_protected:Npn</code> . . . . .	476	<code>\ExpandArgs</code> . . . . .	541, 554, 588, 621
<code>\csname</code> . . . . .	660	<code>\ExplSyntaxOff</code> . . . . .	1117
<b>D</b>			
<code>\DeclareDocumentCommand</code> . . . . .	541	<code>\ExplSyntaxOn</code> . . . . .	8
<code>\DeclareMathEnvironment</code> . . . . .	8	<b>F</b>	
<code>\DeclareRobustCommand</code> . . . . .	653	<code>\fi</code> . . . . .	418, 793, 852, 855, 906, 934, 937, 943, 948, 949, 957, 961, 1109, 1110
<code>\def</code> . . . . .	656, 657, 805, 845, 909, 965, 1099	file commands:	
		<code>\file_if_exist:nTF</code> . . . . .	141
		<code>\file_input:n</code> . . . . .	144
<b>G</b>			
<code>\global</code> . . . . .	943, 944, 949, 950, 961, 962, 1066, 1070	<code>\grabaformulapartandmayberestart</code> . . . . .	1081, 1107
<code>\grabaformulapartandstart</code> . . . . .	410, 1059		

group commands:	keys commands:
\group_insert_after:N . . . . . 792	\keys_define:nm . . . . . 100, 190, 537
	\keys_set:nm . . . . . 550, 584, 617
<b>H</b>	
\hbox . . . . . 808, 928, 932	<b>L</b>
hbox commands:	\lastskip . . . . . 513
\hbox_set:Nn . . . . . 131	\LaTeXe . . . . . 7
	\leavevmode . . . . . 806
<b>I</b>	legacy commands:
\iffalse . . . . . 852	\legacy_if:nTF . . . . . 460, 672
\ifnum . . . . . 852	\legacy_if_p:n . . . . . 484
\iftrue . . . . . 787, 909	\legacy_if_set_false:n . . . . . 673
\ignorespacesafterend . . . . 843, 856, 907	\let . . . . . 812, 813, 909
int commands:	\ltxlabmathdate . . . . . 4
\int_compare:nNnTF . . . . . 186, 1021	\ltxlabmathversion . . . . . 5
\int_eval:n . . . . . 1014, 1028	
\int_gincr:N . . . . . 52, 55, 353, 368, 371	<b>M</b>
\int_new:N . . . . . 42, 43, 44, 45, 46, 177, 178	math commands:
\int_set:Nn . . . . . 184, 791	\math_processor:n . . . . . 3, 475, 475
\int_use:N . . . . .	\math_register_env:n . . . . . 3,
. . . . . 87, 163, 165, 167, 169, 171, 179	547, 705, 714, 715, 717, 720, 800, 801
\int_value:w . . . . . 1028	\math_register_env:nm . . . . .
int internal commands:	. . . . . 3, 547, 547, 706, 709
\g_math_AF_attached_int . . . . . 10	\math_register_halign_env:nm 581,
\g_math_AF_found_int . . . . . 10	911, 912, 913, 914, 915, 916, 917,
\g_math_AF_total_int . . . . . 10	918, 919, 920, 921, 922, 923, 1114, 1115
\g_math_math_total_int . . . . .	\math_register_odd_env:nm . . . . . 614
. . . . . 10, 44, 87, 167, 353	math internal commands:
\g_math_mathchoice_int . . . . . 14, 177	\_math_AF_html_reader:w . . . . 64, 138
\g_math_mathml_AF_attached_int . . . . .	\_math_AF_html_reader_verb:w 70, 72
. . . . . 10, 46, 171, 371	\_math_AF_mml:nmnn . . . . . 49, 49, 74
\g_math_mathml_AF_found_int . . . . .	\_math_AF_process_mathml_files:
. . . . . 10, 45, 169, 368	. . . . . 128, 129, 176
\g_math_mathml_int . . . . . 10, 43, 55, 165	\_math_amsmath_align@:nm . . . . . 830
\g_math_mathml_total_int . . . . .	\_math_amsmath_gather@:n . . . . . 830
. . . . . 10, 42, 52, 163	\_math_amsmath_multline@:n . . . . 830
\l_math_mathstyle_int . . . . . 177, 184	\_math_env_end: . . . . . 565, 599, 632, 710
\g_math_mml_int . . . . . 10	\_math_env_forward:w . . . . . 544, 710, 710
\g_math_mml_total_int . . . . . 10	\_math_equation_begin: . . . . . 712
\intertext . . . . . 8	\_math_equation_end: . . . . . 712
iow commands:	\_math_equation_star_begin: . . . 712
\iow_close:N . . . . . 122	\_math_equation_star_end: . . . . 712
\iow_new:N . . . . . 107	\_math_grab_dollar:w . . . . . 477, 477, 780
\iow_newline: . . . . .	\_math_grab_dollardollar:w . . . . .
. . . . . 84, 86, 88, 90, 92, 94, 96, 115	. . . . . 496, 496, 797
\iow_now:Nn . . . . . 82, 112, 121	\_math_grab_eqn:w . . . . . 530, 530, 745, 748
\iow_open:Nn . . . . . 108	\_math_grab_inline:w . . . . . 521, 521, 732
\iow_term:n . . . . .	\_math_m@th: . . . . . 816, 816, 820
. . . . . 161, 162, 163, 165, 167, 169, 171	\_math_process:nm . . . . . 458, 458,
iow internal commands:	467, 482, 501, 567, 601, 634, 711, 767
\g_math_writedummy_iow . . . . .	\_math_process_auxi:nm 458, 464, 468
. . . . . 82, 105, 107, 108, 112, 121, 122	\_math_process_auxii:nm . . . . .
	. . . . . 458, 472, 474, 476
<b>K</b>	\_math_split:n . . . . . 1024, 1045
\kern . . . . . 953	

<code>\_math_split_at_nl:NN</code> .....	pdffile commands:
..... 972, 972, 1060, 1082	<code>\pdffile_embed_stream:nnN</code> ..... 57
<code>\_math_split_at_nl_aux:nnNN</code> 975, 987	pdfmanagement commands:
<code>\_math_split_at_nl_first:w</code> .....	<code>\pdfmanagement_add:nnn</code> ..... 59
..... 974, 977, 1046	<code>\penalty</code> ..... 515
<code>\_math_split_check_count_-</code>	Plugs:
<code>begins:nnnn</code> ..... 1009, 1019	<code>actual+source</code> ..... 290
<code>\_math_split_chk_if_begin:ww</code> ...	<code>alt+source</code> ..... 305
..... 982, 993	<code>default</code> ..... 214, 236, 245, 323, 407
<code>\_math_split_cleanup_begin_q_-</code>	<code>MC</code> ..... 206
<code>nil:w</code> ..... 1000, 1005	<code>On</code> ..... 77
<code>\_math_split_collect_one_end:w</code> .	<code>single</code> ..... 422
..... 999, 1007, 1029	<code>test-mathml</code> ..... 346
<code>\_math_split_count_begins:n</code> ...	<code>\postdisplaypenalty</code> ..... 29, 515, 791
..... 1010, 1013	property commands:
<code>\_math_split_count_begins:w</code> ...	<code>\property_new:nnnn</code> ..... 179
..... 1014, 1016, 1017	<code>\property_record:nn</code> ..... 185
<code>\_math_split_final_cleanup:nn</code> ..	<code>\property_ref:nn</code> ..... 186
..... 1023, 1032	<code>\ProvidesFile</code> ..... 3
<code>\_math_split_final_cleanup:w</code> ...	
..... 1034, 1038, 1042	<b>Q</b>
<code>\_math_split_guard:n</code> 1024, 1035, 1038	quark commands:
<code>\_math_tag_dollardollar_-</code>	<code>\q_nil</code> .....
<code>display_end:</code> ..... 507, 792	..... 974, 982, 994, 1005, 1014, 1035, 1046
<code>\_math_tag_if_mathstyle:nn</code> ....	<code>\quark_if_nil:nTF</code> 979, 996, 1017, 1040
..... 180, 180, 189	
<code>\mathchoice</code> ..... 13, 14	<b>R</b>
<code>mathstyle</code> ..... 177	<code>\RegisterMathEnvironment</code> ..... 3, 8, 547
<code>\maybestartnewformulatag</code> .....	<code>\relax</code> ..... 938
..... 941, 959, 967, 1055, 1099	<code>\RenewDocumentEnvironment</code> .....
<code>\MaybeStop</code> ..... 8	..... 554, 588, 621, 718, 721
<code>MC (plug)</code> ..... 206	<code>\renewenvironment</code> ..... 835, 858, 863,
<code>\meaning</code> ..... 427, 1061, 1062, 1083, 1084	868, 873, 878, 883, 888, 893, 898, 904
<code>\mml</code> ..... 11, 138	<code>\RequirePackage</code> ..... 7, 9, 802
mode commands:	<code>\result</code> ..... 1057,
<code>\mode_if_math:TF</code> ..... 727, 737, 760	1060, 1061, 1073, 1082, 1083, 1093
<code>\mode_if_vertical:TF</code> ..... 670	<code>\resulttitle</code> ..... 1071, 1075, 1091, 1094
<b>N</b>	<b>S</b>
<code>\NewDocumentCommand</code> ..... 708, 1049	scan commands:
<code>\newif</code> ..... 1056	<code>\s_mark</code> .... 982, 994, 1035, 1038, 1042
<code>\nonumber</code> ..... 851, 937, 948	<code>\s_stop</code> ..... 974,
<code>\normalfont</code> ..... 928, 932	977, 983, 994, 1000, 1007, 1029, 1046
	<code>\show</code> ..... 1052
<b>O</b>	<code>\ShowTagging</code> .. 510, 668, 669, 693, 697, 701
<code>\omit</code> ..... 952	<code>single (plug)</code> ..... 422
<code>On (plug)</code> ..... 77	skip commands:
	<code>\skip_new:N</code> ..... 15
<b>P</b>	<code>\skip_set:Nn</code> ..... 789, 790
para commands:	<code>\skip_vertical:n</code> ..... 516
<code>\para_raw_end:</code> ..... 511, 698	skip internal commands:
pdf commands:	<code>\l_math_tmpa_skip</code> ... 9, 15, 513, 516
<code>\pdf_name_from_unicode_e:n</code> .... 136	socket commands:
pdffdict commands:	<code>\socket_assign_plug:nn</code> .....
<code>\pdffdict_put:nnn</code> ..... 56, 133, 134	..... 320, 344, 345, 420, 421, 437,

438, 439, 440, 441, 442, 443, 444, 448, 449, 450, 451, 452, 453, 454, 455	
<code>\socket_new:nn</code> .....	76, 202, 203, 204, 205, 232, 233, 234, 235, 289, 321, 322, 405, 406, 434
<code>\socket_new_plug:nnn</code> .....	77, 206, 210, 214, 225, 236, 240, 245, 255, 290, 305, 323, 339, 349, 399, 407, 411, 422, 430
<code>\socket_use:n</code> ..	218, 219, 221, 223, 229, 230, 249, 250, 251, 253, 259, 260, 381, 488, 491, 492, 502, 514, 700
<code>\socket_use:nn</code> .....	489, 503, 691
Sockets:	
<code>tagsupport/math/content</code> .....	289
<code>tagsupport/math/display/begin</code> ..	232
<code>tagsupport/math/display/end</code> ...	232
<code>tagsupport/math/display/formula/begin</code> .....	232
<code>tagsupport/math/display/formula/end</code> .....	232
<code>tagsupport/math/end</code> .....	434
<code>tagsupport/math/inline/begin</code> ..	202
<code>tagsupport/math/inline/end</code> ...	202
<code>tagsupport/math/inline/formula/begin</code> .....	202
<code>tagsupport/math/inline/formula/end</code> .....	202
<code>tagsupport/math/mathml/write</code> ...	76
<code>tagsupport/math/struct/begin</code> ..	321
<code>tagsupport/math/struct/end</code> ...	321
<code>tagsupport/math/substruct/begin</code> ..	405
<code>tagsupport/math/substruct/end</code> ..	405
<code>\space</code> .....	4, 5, 1071, 1091
split commands:	
<code>\split:n</code> .....	1051
<code>\splitnl</code> .....	1049
splitnl internal commands:	
<code>\_splitnl_aux:nnNN</code> .....	1053
str commands:	
<code>\c_backslash_str</code> .....	87
<code>\str_if_eq:nnTF</code> .....	680
<code>\str_mdfive_hash:n</code> .....	346, 355
<code>\str_replace_all:Nnn</code> .....	80, 81
<code>\str_set:Nn</code> .....	79
str internal commands:	
<code>\l_math_tmpa_str</code> ...	9, 79, 80, 81, 89
sys commands:	
<code>\c_sys_jobname_str</code> .....	48, 110
<b>T</b>	
<code>\tabcolsep</code> .....	812
tag commands:	
<code>\tag_if_active:TF</code> .....	23, 457, 678
<code>\tag_mc_begin:n</code> .....	428
<code>\tag_mc_begin_pop:n</code> .....	213
<code>\tag_mc_end:</code> .....	25, 433
<code>\tag_mc_end_push:</code> .....	25, 209
<code>\tag_start:</code> .....	187
<code>\tag_stop:</code> .....	187, 909
<code>\tag_struct_begin:n</code> .....	4, 330, 385
<code>\tag_struct_end:</code> .....	28, 342, 403
tag internal commands:	
<code>\_tag_check_para_end_show:nn</code> ...	27
<code>\_tag_gincr_para_end_int:</code> .....	26
<code>\l_tag_math_alt_bool</code> ..	10, 41, 193, 364
<code>\_tag_math_disable:</code> .....	435, 435
<code>\_tag_math_enable:</code> ...	446, 446, 457
<code>\g_tag_math_mathml_AF_bool</code> ....	10, 39, 104, 145, 152
<code>\l_tag_math_mathml_AF_bool</code> ....	10, 38, 197, 369, 390
<code>\l_tag_math_mathml_files_clist</code> .	11, 47, 48, 139, 192
<code>\l_tag_math_mathml_pane_bool</code> ...	10, 40, 58, 194
<code>\l_tag_math_texsource_AF_bool</code> ..	10, 36, 199, 327, 382
<code>\l_tag_math_texsource_pane_bool</code> .....	10, 37, 196
<code>\l_tag_para_main_tag_tl</code> .....	680
<code>\g_tag_struct_tag_tl</code> .....	676, 680
<code>\_tag_tool_close_P:</code> ..	21, 21, 239, 689
<code>\tagmcbegin</code> .....	931, 933, 1078, 1097
<code>\tagmcbend</code> .....	415, 930, 933, 1105
<code>\tagpdfparaOff</code> .....	692
<code>\tagpdfparaOn</code> .....	512, 699
<code>\tagpdfsetup</code> .....	32
<code>\tagstructbegin</code> .....	930, 1072, 1092
<code>\tagstructend</code> .....	417, 933, 1106
<code>tagsupport/math/content (socket)</code> ...	289
<code>tagsupport/math/display/begin (socket)</code> .....	232
<code>tagsupport/math/display/end (socket)</code> ..	232
<code>tagsupport/math/display/formula/begin</code> (socket) .....	232
<code>tagsupport/math/display/formula/end</code> (socket) .....	232
<code>tagsupport/math/end (socket)</code> .....	434
<code>tagsupport/math/inline/begin (socket)</code> ..	202
<code>tagsupport/math/inline/end (socket)</code> .	202
<code>tagsupport/math/inline/formula/begin</code> (socket) .....	202
<code>tagsupport/math/inline/formula/end</code> (socket) .....	202
<code>tagsupport/math/mathml/write (socket)</code> ..	76
<code>tagsupport/math/struct/begin (socket)</code> ..	321
<code>tagsupport/math/struct/end (socket)</code> .	321



tagsupport/math/substruct/begin (socket) .....	405	$\m@th$ ..	7, 9, 22, 30, 462, <u>816</u> , 928, 932
tagsupport/math/substruct/end (socket) .....	<u>405</u>	$\make@display@tag$ .....	939, 955
test-mathml (plug) .....	<u>346</u>	$\math@cr$ .....	842, 846
TEX and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:		$\maxfields@$ .....	951
$\@badmath$ .....	728, 739, 752, 755	$\measuring@true$ .....	909
$\@currenvir$ .....	657	$\multline@$ .....	830
$\@currenvline$ .....	658	$\on@line$ .....	658
$\@doendpe$ .....	518	$\place@tag$ .....	956
$\@eha$ .....	656	$\rendmultline@$ .....	906
$\@endpefalse$ .....	663	$\reserved@a$ .....	656, 657, 664
$\@endpetrue$ .....	702	$\restore@math@cr$ .....	965, 970
$\@ensuredmath$ .....	22, 764, 768	$\restorealignstate@$ .....	849
$\@eqnswtrue$ .....	943, 961	$\row@$ .....	944, 950
$\@execute@begin@hook$ .....	659	$\setboxz@h$ .....	955
$\@ifundefined$ .....	655	$\split@tag$ .....	<u>30</u>
$\@ignorefalse$ .....	661	$\st@rredfalse$ .....	859, 869, 884, 894
$\@iiiparbox$ .....	7	.....	836, 864, 874, 879, 889, 899, 904
$\@kernel@after@begindocument$ ..	826	$\start@align$ .....	859,
$\@kernel@before@begindocument$ ..	712, 741, 822, 833	864, 869, 874, 879, 884, 889, 894, 899	
$\@kernel@math@registered@begin$ ..	568, 602, 635, 667	$\start@gather$ .....	836
$\@kernel@math@registered@end$	646, 696	$\start@multline$ .....	904
$\@latex@error$ .....	656	$\strut@$ .....	955
$\@lign$ .....	955	$\tag@true$ .....	949
$\@namedef$ .....	925, 936, 947, 965	$\totwidth@$ .....	842, 846
$\@ne$ .....	869, 874, 884, 889, 944, 950	$\tw@$ .....	879, 894, 899
$\@subformulasfalse$ .....	1066	$\z@$ .....	859, 864, 962
$\@subformulastrue$ .....	1070	tex commands:	
$\@tabarray$ .....	814	$\tex\_everydisplay:D$ .....	784, 786
$\@tabular$ .....	803, 805	$\tex\_everymath:D$ .....	774, 776
$\@xp$ .....	906	$\tex\_the:D$ .....	776, 786
$\@add@amps$ .....	951	$\text$ .....	14
$\@align@$ .....	830	tl commands:	
$\@alignsep@$ .....	953	$\@c\_space\_tl$ .....	87,
$\@black@$ .....	842, 846	163, 165, 167, 169, 171, 267, 269, 271	
$\@col@sep$ .....	812	$\@tl\_clear:N$ .....	329, 384
$\@common@align@ending$ ..	845, 861,	$\@tl\_const:Nn$ .....	274
866, 871, 876, 881, 886, 891, 896, 901		$\@tl\_gput\_right:Nn$ .....	
$\@d@llarbegin$ .....	812	.....	712, 741, 822, 826, 833
$\@d@llarend$ .....	813	$\@tl\_gset:Nn$ .....	470, 471, 989, 990
$\@gather@$ .....	830	$\@tl\_gset\_eq:NN$ .....	61
$\@if@eqnsw$ .....	949	$\@tl\_if\_blank:nTF$ .....	480, 499, 524
$\@if@subformulas$ .....	416, 1056, 1100	$\@tl\_if\_empty:NTF$ .....	1063, 1085, 1103
$\@ifingather@$ .....	848	$\@tl\_if\_eq:NNTF$ .....	277
$\@ifmeasuring@$ .....	3, 7, 909, 927, 1101	$\@tl\_if\_eq:NnTF$ .....	357
$\@ifst@rred$ .....	937, 943, 948, 961	$\@tl\_if\_exist:NTF$ .....	53, 105, 366
$\@iftag@$ .....	954	$\@tl\_if\_in:NnTF$ .....	803
$\@iftagsleft@$ .....	906	$\@tl\_if\_in:nTF$ .....	462
$\@lendmultline@$ .....	906	$\@tl\_new:N$ ..	12, 13, 14, 16, 17, 18, 19,
$\@lineht@$ .....	962	20, 31, 60, 262, 273, 347, 348, 536, 1057	
$\@m@ne$ .....	884, 889, 894, 899	$\@tl\_set:Nn$ .....	263, 275, 294, 298,
		302, 303, 309, 313, 317, 318, 354,	
		359, 362, 365, 549, 583, 616, 973, 1051	
		$\@tl\_set\_eq:NN$ .....	328, 356, 383

