

The `latex-lab-math` code*

Frank Mittelbach, Joseph Wright, L^AT_EX Project

v0.6b 2024-08-14

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

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

*

5	The Implementation	8
5.1	File declaration	8
5.2	Setup	9
5.3	Data structures	9
5.4	Tagging tools	10
5.5	Code related to AF	10
5.6	Mathstyle detection	16
5.7	Tagging options	17
5.8	Sockets	17
5.8.1	Main inline math sockets	17
5.8.2	Main display math sockets	18
5.8.3	Internal sockets	19
5.9	Interface commands	23
5.10	Content grabbing	24
5.11	Token-by-token inline grabbing	26
5.12	Marking math environments	28
5.13	Document commands	32
5.14	<code>\everymath</code> and <code>\everydisplay</code>	34
5.15	Modifying kernel environments	34
5.16	Disable math grabbing in the <code>begindocument</code> hook	35
5.17	Modifying <code>amsmath</code>	35
	Index	41

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,

$$\left[a \in A \text{ for all } a \right]$$

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

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^AT_EX 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^AT_EX 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 the files `\jobname-mathml.html` and/or `\jobname-luamml-mathml.html` are found and if they contains a suitable MathML snippet for the current formula. If the files contain more than one suitable snippet (as identified by the hash) the first one is used. `\jobname-luamml-mathml.html` is automatically generated (see below section 3.4) and read after `\jobname-mathml.html`. This means that `\jobname-mathml.html` can contain improved versions of a 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_bl substructure is ignored if an MathML is used (see 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^AT_EX 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"`
1

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 Automatic mathml creation with luamml

If `lualatex` is used, if the package `unicode-math` is used and if the package `luamml` is found² then it will automatically generate the file `\jobname-luamml-mathml.html` with `mathml` representations of all math formulas. The generation of the file can be suppressed with `math/mathml/luamml=false`. The package is still quite experimental and the output should be checked. The `\jobname-luamml-mathml.html` file may be previewed in a browser although you may need to add additional `css` or `javascript` declarations to enable browser support for all `mathml` constructs. The file is then used in subsequent compilations and works also with `pdflatex`.

3.5 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,

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

¹But it is probably not needed and only blows up the PDF.

²Until it is on CTAN and it must be installed from <https://github.com/zauguin/luamml>.

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 `\)` `\[`), 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 `\[` 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 `\[` to fall back to `equation*` doesn't work if `equation*` is a grabbing environment.
6. The behaviour of `\[...]` is faulty. See above.

4.2 Fake math

In a number of places in \LaTeX 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 \LaTeXe 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 reenact 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 `\LaTeXe` 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 `lcmd` [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

Change description here?

```

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

```

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

storage tl:
~~`\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
16 \str_new:N \l__math_tmpa_str

```

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

```

17 \tl_new:N \l__math_content_alt_tl
18 \tl_new:N \l__math_content_actual_tl
19 \tl_new:N \l__math_content_AF_source_tl
20 \tl_new:N \l__math_content_AF_source_tmpa_tl
21 \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.

```
22 \cs_new_protected:Npn \__tag_tool_close_P:
23 {
24   \tag_if_active:T
25   {
26     \tag_mc_end: %end P-chunk, should perhaps be \tag_mc_end_push: ...
27     \__tag_gincr_para_end_int:
28     \__tag_check_para_end_show:nn{red}{} %debug: show para
29     \tag_struct_end:
30   }
31 }
```

(End of definition for __tag_tool_close_P:.)

We add also an attribute.

```
32 \tl_new:N\l__math_attribute_class_tl
33 \tagpdfsetup
34   {role/new-attribute = {inline}    {/O /Layout /Placement/Inline},
35   {role/new-attribute = {display}  {/O /Layout /Placement/Block},
36   }
```

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
\g__tag_math_luamml_bool
```

```
37 \bool_new:N\l__tag_math_texsource_AF_bool
38 \bool_new:N\l__tag_math_texsource_pane_bool
39 \bool_new:N\l__tag_math_mathml_AF_bool
40 \bool_new:N\g__tag_math_mathml_AF_bool
41 \bool_new:N\l__tag_math_mathml_pane_bool
42 \bool_new:N\l__tag_math_alt_bool
43 \bool_new:N\g__tag_math_luamml_bool
44 \bool_gset_true:N \g__tag_math_luamml_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).

```

45 \int_new:N\g__math_mathml_total_int
46 \int_new:N\g__math_mathml_int
47 \int_new:N\g__math_math_total_int
48 \int_new:N\g__math_mathml_AF_found_int
49 \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. If luatex is detected with also check the luamml file.

```

50 \clist_new:N\l__tag_math_mathml_files_clist
51 \clist_put_right:Ne\l__tag_math_mathml_files_clist
52   {\c_sys_jobname_str-mathml,\c_sys_jobname_str-luamml-mathml}

```

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

```

\__math_AF_mml:nnnn

```

```

53 \cs_new_protected:Npn \__math_AF_mml:nnnn #1 #2 #3 #4
54   {%#1 number, #2 tex source for debugging, #3 hash, #4 mathml
55   {
56     \int_gincr:N \g__math_mathml_total_int

```

mathml with the same hash should be included only once:

```

57     \tl_if_exist:cF { g__math_mathml_#3_tl }
58     {
59       \int_gincr:N \g__math_mathml_int

```

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

```

60         \pdfdict_put:nne {l_pdffile/Filespec} {Desc}{(mathml-#1)}
61         \pdffile_embed_stream:nnN {#4}{mathml-#1.xml}\l__math_tmpa_tl

```

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

```

62         \bool_if:NT \l__tag_math_mathml_pane_bool
63         {\pdfmanagement_add:nne {Catalog/Names}{EmbeddedFiles}{\l__math_tmpa_tl}}
64         \tl_new:c{g__math_mathml_#3_tl}
65         \tl_gset_eq:cN{g__math_mathml_#3_tl}\l__math_tmpa_tl
66     }
67 }

```

(End of definition for `_math_AF_mml:nnnn`.)

The html reader.

```
68 \cs_new_protected:Npn \_math_AF_html_reader:w#1</h2>#2<p>#3</p>#4<p>#5</p>#6<math{
69 \begingroup
70 \char_set_catcode_other:N\{
71 \char_set_catcode_other:N\}
72 \char_set_catcode_other:N\#
73 \char_set_catcode_other:N\%
74 \_math_AF_html_reader_verb:w{#1}{#3}{#5}<math
75 }
76 \cs_new_protected:Npn\_math_AF_html_reader_verb:w#1#2#3#4~</div>{
77 \endgroup
78 \_math_AF_mml:nnnn{#1}{#2}{#3}{#4}
79 }
```

As with `luatex` we write two files we define a few constants for the shared texts.

```
\c_math_mathml_write_init_tl
\l_math_mathml_write_before_tl
\c_math_mathml_write_after_tl
\c_math_mathml_write_final_tl
80 \tl_const:Nn \c_math_mathml_write_init_tl
81 {
82 <!DOCTYPE~html>
83 \iow_newline:
84 <html>
85 \iow_newline:
86 }
87 \tl_new:N \l_math_mathml_write_before_tl
88 \tl_const:Nn \c_math_mathml_write_after_tl
89 {
90 \iow_newline:
91 </div>
92 \iow_newline:
93 }
94 \tl_const:Nn \c_math_mathml_write_final_tl
95 {
96 </html>
97 }
```

(End of definition for `\c_math_mathml_write_init_tl` and others.)

`h/mathml/write/prepare` (*socket*) To prepare the hash and the starting command we use a socket, so that both the dummy and `luamml` can make use of it.

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

On (*plug*)

```
99 \socket_new_plug:nnn{tagsupport/math/mathml/write/prepare}{0n}
100 {
101 \str_set:NV\l_math_tmpa_str\l_math_content_AF_source_tl
102 \str_replace_all:Nnn\l_math_tmpa_str{&}{&amp;}
103 \str_replace_all:Nnn\l_math_tmpa_str{<}{&lt;}
104 \tl_set:Nn \l_math_mathml_write_before_tl
105 {
106 <div>
107 \iow_newline:
108 <h2>\c_backslash_str mml\c_space_tl \int_use:N \g_math_math_total_int </h2>
```

```

109     \iow_newline:
110     <p>\l__math_tmpa_str</p>
111     \iow_newline:
112     <p>\l__math_content_hash_tl </p>
113     \iow_newline:
114   }
115 }

```

With luatex we automatically generate mathml with luamml if the package can be loaded and unicode-math is detected. We start the process in the `begindocument/end` hook so that the reading from a previous compilation can happen before!

```

116 \sys_if_engine luatex:T
117 {
118   \file_if_exist:nT { luamml.sty }
119   {
120     \RequirePackage { luamml }
121     \AddToHook{begindocument/end}
122     {
123       \bool_if:NT \g__tag_math_luamml_bool
124       {
125         \@ifpackageloaded { unicode-math }
126         { \__math_luamml_activate_write: }
127         { \msg_warning:nn { tag }{ unicode-math-missing } }
128       }
129     }
130   }
131 }
132 \msg_new:nnn { tag }{ unicode-math-missing }
133 {
134   The~package~unicode-math-is~missing\\
135   luamml~will~not~create~an~MathML~file.\\
136   To~avoid~this~warning~load~unicode-math~or~disable~luamml\\
137   with~\tl_to_str:n{\tagpdfsetup{math/mathml/luamml=false}}
138 }
139 \cs_new_protected:Npn \__math_luamml_activate_write:
140 {

```

to avoid that nothing is written in the first run, we must activate the sockets:

```

141   \bool_gset_true:N\g__tag_math_mathml_AF_bool
142   \AssignSocketPlug{tagsupport/math/struct/begin}{test-mathml}
143   \AssignSocketPlug{tagsupport/math/struct/end}{test-mathml}
144   \AssignSocketPlug{tagsupport/math/substruct/begin}{single}
145   \AssignSocketPlug{tagsupport/math/substruct/end}{single}
146   \int_set:Nn \l__luamml_pretty_int { 5 }
147   \RegisterFamilyMapping\symsymbols{oms}
148   \RegisterFamilyMapping\symletters{oml}
149   \AssignSocketPlug{tagsupport/math/mathml/write/prepare}{0n}
150   \iow_new:N \g__math_luamml_iow
151   \iow_open:Nn \g__math_luamml_iow {\c_sys_jobname_str-luamml-mathml.html}
152   \iow_now:Ne \g__math_luamml_iow { \c__math_mathml_write_init_tl }
153   \cs_new:Npn \__math_luamml_output_hook:n ##1
154   {
155     \tl_if_empty:NF \l__math_mathml_write_before_tl
156     {

```

```

157     \iow_now:Ne \g__math_luaamml_iow
158     {
159         \l__math_mathml_write_before_tl
160         ##1
161         \c__math_mathml_write_after_tl
162     }
163 }
164 }
165 \__luaamml_register_output_hook:N \__math_luaamml_output_hook:n

```

this starts the process and set flag to 3, not really needed, as it is the default but set here for clarity:

```

166 \luaamml_flag_process:

```

At the end of the document we must finish and close the file:

```

167 \AddToHook{enddocument/afterlastpage}
168 {
169     \iow_now:Ne \g__math_luaamml_iow
170     { \c__math_mathml_write_final_tl }
171     \iow_close:N \g__math_luaamml_iow
172 }
173 }

```

And now a key to deactivate luaamml

```

174
175 \keys_define:nn { __tag / setup }
176 {
177     math/mathml/luaamml .bool_gset:N = \g__tag_math_luaamml_bool,
178     math/mathml/luaamml .usage:n=preamble
179 }

```

`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

```

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

```

On (plug)

```

181 \socket_new_plug:nnn{tagsupport/math/mathml/write}{0n}
182 {
183     \iow_now:Ne \g__math_writedummy_iow
184     {
185         \l__math_mathml_write_before_tl
186         <math></math>
187         \c__math_mathml_write_after_tl
188     }
189 }

```

And now a key to activate the socket.

```

190
191 \keys_define:nn { __tag / setup }
192 {
193     math/mathml/write-dummy .code:n =
194     {
195         \bool_gset_true:N \g__tag_math_mathml_AF_bool
196         \tl_if_exist:NF\g__math_writedummy_iow

```

```

197     {
198       \iow_new:N \g__math_writedummy_iow
199       \iow_open:Nn \g__math_writedummy_iow
200         {
201           \c_sys_jobname_str-mathml-dummy.html
202         }
203       \iow_now:Ne \g__math_writedummy_iow
204         {
205           \c__math_mathml_write_init_tl
206         }
207       \AssignSocketPlug {tagsupport/math/mathml/write/prepare}{0n}
208       \AssignSocketPlug {tagsupport/math/mathml/write}{0n}
209       \AddToHook{enddocument/afterlastpage}
210         {
211           \iow_now:Ne \g__math_writedummy_iow
212             { \c__math_mathml_write_final_tl }
213           \iow_close:N \g__math_writedummy_iow
214         }
215     }
216   },
217   math/mathml/write-dummy .usage:n=preamble
218 }

```

_math_AF_process_mathml_files:

```

219 \box_new:N\l__math_tmpa_box
220 \cs_new_protected:Npn \_math_AF_process_mathml_files:
221 {
222   \hbox_set:Nn \l__math_tmpa_box
223     {
224       \pdfdict_put:nnn { l_pdffile/Filespec }{AFRelationship} { /Supplement }
225       \pdfdict_put:nne
226         { l_pdffile }{Subtype}
227         { \pdf_name_from_unicode_e:n{application/mathml+xml} }
228       \char_set_catcode_other:N \#
229       \cs_set_eq:NN\mml\_math_AF_html_reader:w
230       \clist_map_inline:Nn \l__tag_math_mathml_files_clist
231         {
232           \file_if_exist:nTF {##1.html}
233             {
234               \typeout{Info:~reading~mathml~file~##1}
235               \file_input:n {##1.html}
236               \bool_gset_true:N\g__tag_math_mathml_AF_bool
237             }
238             {
239               \typeout{Info:~mathml~file~##1~does~not~exist}%info message
240             }
241         }
242     }
243   \bool_if:NT\g__tag_math_mathml_AF_bool
244     {
245       \typeout{Info:~Activating~mathml~support}
246       \AssignSocketPlug{tagsupport/math/struct/begin}{test-mathml}
247       \AssignSocketPlug{tagsupport/math/struct/end}{test-mathml}

```

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

```

248     \AssignSocketPlug{tagsupport/math/substruct/begin}{single}
249     \AssignSocketPlug{tagsupport/math/substruct/end}{single}
250     \AddToHook{enddocument/info}
251     {
252         \iow_term:n{MathML~statistic}
253         \iow_term:n{=====}
254         \iow_term:e{==>~\int_use:N\g__math_mathml_total_int\c_space_tl
255         MathML~fragments~read}
256         \iow_term:e{==>~\int_use:N\g__math_mathml_int\c_space_tl
257         different~MathML~fragments}
258         \iow_term:e{==>~\int_use:N\g__math_math_total_int\c_space_tl
259         math~fragments~found}
260         \iow_term:e{==>~\int_use:N\g__math_mathml_AF_found_int\c_space_tl
261         fitting~MathML~AF~found}
262         \iow_term:e{==>~\int_use:N\g__math_mathml_AF_attached_int\c_space_tl
263         MathML~AF~attached}
264     }
265 }
266 }
267 \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
268 \int_new:N \l__math_mathstyle_int
269 \int_new:N \g__math_mathchoice_int
270 \property_new:n{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
271 \cs_new_protected:Npn \__math_tag_if_mathstyle:nn #1 #2
272   %#1 refers to label
273   %#2 is a number for the mathstyle (typically 0,2,4,6)
274   {
275     \int_set:Nn \l__math_mathstyle_int {#2}
276     \property_record:nn {#1} { mathstyle }
277     \int_compare:nNnTF { \property_ref:nn {#1}{ mathstyle } } = { #2 }
278     { \tag_start: }{ \tag_stop: }
279   }
280 \cs_generate_variant:Nn \__math_tag_if_mathstyle:nn {en}

```

(End of definition for `_math_tag_if_mathstyle:nn`.)

5.7 Tagging options

```
281 \keys_define:nn { __tag / setup }
282   {
283     math/mathml/sources .clist_set:N = \l__tag_math_mathml_files_clist,
284     math/alt/use        .bool_set:N = \l__tag_math_alt_bool,
285     viewer/pane/mathml .bool_set:N = \l__tag_math_mathml_pane_bool,
286     viewer/pane/mathml .initial:n = true,
287     viewer/pane/mathsource .bool_set:N = \l__tag_math_texsource_pane_bool,
288     math/mathml/AF .bool_set:N = \l__tag_math_mathml_AF_bool,
289     math/mathml/AF .initial:n = true,
290     math/tex/AF .bool_set:N = \l__tag_math_texsource_AF_bool,
291     math/tex/AF .initial:n = true
292   }
```

5.8 Sockets

5.8.1 Main inline math sockets

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

```
293 \socket_new:nn {tag_support/math/inline/begin}{0}
294 \socket_new:nn {tag_support/math/inline/end}{0}
295 \socket_new:nn {tag_support/math/inline/formula/begin}{1} %
296 \socket_new:nn {tag_support/math/inline/formula/end}{0}
```

MC (*plug*)

```
297 \socket_new_plug:nnn
298   {tag_support/math/inline/begin}
299   {MC}
300   {\tag_mc_end_push:}
301 \socket_new_plug:nnn
302   {tag_support/math/inline/end}
303   {MC}
304   {\tag_mc_begin_pop:n{}}
```

We probably will want to test different tagging recipes.

default (*plug*)

```
305 \socket_new_plug:nnn
306   {tag_support/math/inline/formula/begin}
307   {default}
308   {
309     \socket_use:n{tag_support/math/content}
310     \socket_use:n{tag_support/math/struct/begin}
```

TODO: does inline math need subformula handling?

```
311     % inner formula if multiple parts (not really implemented yet)
312     \socket_use:n{tag_support/math/substruct/begin}
313     #1
314     \socket_use:n{tag_support/math/end}
```

```

315 }
316 \socket_new_plug:nnn
317 {tagsupport/math/inline/formula/end}
318 {default}
319 {
320   \socket_use:n{tagsupport/math/substruct/end}
321   \socket_use:n{tagsupport/math/struct/end}
322 }

```

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
`port/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!

```

323 \socket_new:nn {tagsupport/math/display/begin}{0}
324 \socket_new:nn {tagsupport/math/display/end}{0}
325 \socket_new:nn {tagsupport/math/display/formula/begin}{1} %
326 \socket_new:nn {tagsupport/math/display/formula/end}{0}

```

default (*plug*)

```

327 \socket_new_plug:nnn
328 {tagsupport/math/display/begin}
329 {default}
330 { \_tag_tool_close_P: }
331 \socket_new_plug:nnn
332 {tagsupport/math/display/end}
333 {default}
334 {
335 }

```

default (*plug*)

```

336 \socket_new_plug:nnn
337 {tagsupport/math/display/formula/begin}
338 {default}
339 {
340   \socket_use:n{tagsupport/math/content}
341   \socket_use:n{tagsupport/math/struct/begin}
342   \socket_use:n{tagsupport/math/substruct/begin}
343   #1
344   \socket_use:n{tagsupport/math/end}
345 }
346 \socket_new_plug:nnn
347 {tagsupport/math/display/formula/end}
348 {default}
349 {
350   \socket_use:n{tagsupport/math/substruct/end}
351   \socket_use:n{tagsupport/math/struct/end}
352 }

```

5.8.3 Internal sockets

`\l__math_content_template_tl`

The default text used as alt or actual text.

```

353 \tl_new:N\l__math_content_template_tl
354 \tl_set:Nn \l__math_content_template_tl
355   {
356     LaTeX~ formula~ starts~
357     \exp_not:N\begin{\g__math_grabbed_env_tl}
358     \c_space_tl
359     \exp_not:V\g__math_grabbed_math_tl
360     \c_space_tl
361     \exp_not:N\end{\g__math_grabbed_env_tl}
362     \c_space_tl LaTeX~ formula~ ends~
363   }
```

`\l__math_texsource_template_tl`

The default text used as texsource

```

364 \tl_new:N\l__math_texsource_template_tl
365 \tl_const:Nn\c__math_inline_env_tl {math}
366 \tl_set:Nn \l__math_texsource_template_tl
367   {
368     \tl_if_eq:NNTF\g__math_grabbed_env_tl\c__math_inline_env_tl
369     {
370       $
371       \exp_not:V\g__math_grabbed_math_tl
372       $
373     }
374     {
375       \exp_not:N\begin{\g__math_grabbed_env_tl}
376       \exp_not:V\g__math_grabbed_math_tl
377       \exp_not:N\end{\g__math_grabbed_env_tl}
378     }
379   }
```

`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`

```

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

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

`actual+source` (*plug*)

```

381 \socket_new_plug:nnn
382   {tagsupport/math/content}
383   {actual+source}
384   {
```

```

385 \tl_set:N\l__math_content_actual_tl
386 {
387   \l__math_content_template_tl
388 }
389 \tl_set:N\l__math_content_AF_source_tl
390 {
391   \l__math_texsource_template_tl
392 }
393 \tl_set:Nn \l__math_content_AF_mathml_tl {}
394 \tl_set:Nn \l__math_content_alt_tl {}
395 }

```

`alt+source` (*plug*)

```

396 \socket_new_plug:nnn
397 {tagsupport/math/content}
398 {alt+source}
399 {
400   \tl_set:N\l__math_content_alt_tl
401   {
402     \l__math_content_template_tl
403   }
404   \tl_set:N\l__math_content_AF_source_tl
405   {
406     \l__math_texsource_template_tl
407   }
408   \tl_set:Nn \l__math_content_AF_mathml_tl {}
409   \tl_set:Nn \l__math_content_actual_tl {}
410 }
411 \socket_assign_plug:nn {tagsupport/math/content}{alt+source}

```

`tagsupport/math/struct/begin` (*socket*) For the main structure we use a socket too. This allow e.g. to create a special one
`tagsupport/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`

```

412 \socket_new:nn {tagsupport/math/struct/begin}{0}
413 \socket_new:nn {tagsupport/math/struct/end}{0}

```

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

```

414 \socket_new_plug:nnn
415 {tagsupport/math/struct/begin}
416 {default}
417 {
418   \bool_if:NTF\l__tag_math_texsource_AF_bool
419   { \tl_set_eq:NN \l__math_content_AF_source_tmpa_tl \l__math_content_AF_source_tl }
420   { \tl_clear:N \l__math_content_AF_source_tmpa_tl }
421   \tag_struct_begin:n
422   {
423     tag=Formula,
424     texsource = \l__math_content_AF_source_tmpa_tl,
425     title-o = \g__math_grabbed_env_tl,
426     actualtext = \l__math_content_actual_tl,
427     alt = \l__math_content_alt_tl
428   }
429 }

```

```

430 \socket_new_plug:nnn
431   {tagsupport/math/struct/end}
432   {default}
433   { \tag_struct_end: }
434
435 \socket_assign_plug:nn {tagsupport/math/struct/begin}{default}
436 \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`

```

437 \cs_generate_variant:Nn \str_mdfive_hash:n {o}
438 \tl_new:N\l__math_content_hash_tl

```

we need to save the grabbed math:

```

439 \tl_new:N\l__math_grabbed_math_tl

```

the socket definition

```

440 \socket_new_plug:nnn
441   {tagsupport/math/struct/begin}
442   {test-mathml}
443   {
444     \int_gincr:N\g__math_math_total_int
445     \tl_set:N\l__math_content_hash_tl
446       {\str_mdfive_hash:o { \l__math_content_AF_source_tl }}
447     \tl_set_eq:NN\l__math_grabbed_math_tl\g__math_grabbed_math_tl
448     \tl_if_eq:NnTF\g__math_grabbed_env_tl {math}
449     {
450       \tl_set:Nn\l__math_attribute_class_tl{inline}
451     }
452     {
453       \tl_set:Nn\l__math_attribute_class_tl{display}
454     }
455     \bool_if:NF\l__tag_math_alt_bool
456     { \tl_set:Nn \l__math_content_alt_tl{} }

```

debugging option. TODO: hide in debug key.

```

457   \tl_if_exist:cTF { g__math_mathml_ \l__math_content_hash_tl _tl }
458   {
459     \int_gincr:N\g__math_mathml_AF_found_int
460     \bool_if:NTF \l__tag_math_mathml_AF_bool
461     {
462       \int_gincr:N\g__math_mathml_AF_attached_int
463       \typeout {Inserting~mathml~with~Hash~\l__math_content_hash_tl}
464     }
465     {
466       \typeout {Ignoring~mathml~with~Hash~\l__math_content_hash_tl}
467     }
468   }
469   {
470     \typeout{WARNING:~mathml~missing~for~hash~\l__math_content_hash_tl}
471   }
472   \socket_use:n {tagsupport/math/mathml/write/prepare}
473   \socket_use:n {tagsupport/math/mathml/write} % write hash if request

```

```

474 \bool_if:NTF\l__tag_math_texsource_AF_bool
475 { \tl_set_eq:NN \l__math_content_AF_source_tmpa_tl \l__math_content_AF_source_tl }
476 { \tl_clear:N \l__math_content_AF_source_tmpa_tl }
477 \tag_struct_begin:n
478 {
479   tag=Formula,
480   attribute-class=\l__math_attribute_class_tl, %
481   AFref          =
482   \bool_if:NT\l__tag_math_mathml_AF_bool
483   {
484     \cs_if_exist_use:c {g__math_mathml_ \l__math_content_hash_tl _tl}
485   },
486   texsource      = \l__math_content_AF_source_tmpa_tl, % should be after mathml AF!
487   title-o        = \g__math_grabbed_env_tl, %
488   alt            = \l__math_content_alt_tl
489 }
490 }

```

not really needed but looks more symmetric:

```

491 \socket_new_plug:nnn
492 {tagsupport/math/struct/end}
493 {test-mathml}
494 {
495   \tag_struct_end:
496 }

```

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

```

497 \socket_new:nn {tagsupport/math/substruct/begin}{0}
498 \socket_new:nn {tagsupport/math/substruct/end}{0}

```

default (plug)

```

499 \socket_new_plug:nnn
500 {tagsupport/math/substruct/begin}
501 {default}
502 { \grabaformulapartandstart }
503 \socket_new_plug:nnn
504 {tagsupport/math/substruct/end}
505 {default}
506 {
507   \tagmcentd
508   \if@subformulas
509   \tagstructend
510   \fi
511 }
512 \socket_assign_plug:nn {tagsupport/math/substruct/begin}{default}
513 \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:

```

514 \socket_new_plug:nnn
515 {tagsupport/math/substruct/begin}
516 {single}
517 {
518   \typeout{====>subpart~splitting~deactivated}
519   \typeout{====>grabbed~math=\meaning\g__math_grabbed_math_tl}

```

```

520     \tag_mc_begin:n{ }
521   }
522   \socket_new_plug:nnn
523     {tagsupport/math/substruct/end}
524     {single}
525     { \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.

```

526   \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

```

527   \cs_new_protected:Npn \__tag_math_disable:
528     {
529       \socket_assign_plug:nn {tagsupport/math/inline/begin}{noop}
530       \socket_assign_plug:nn {tagsupport/math/inline/end}{noop}
531       \socket_assign_plug:nn {tagsupport/math/inline/formula/begin}{identity}
532       \socket_assign_plug:nn {tagsupport/math/inline/formula/end}{noop}
533       \socket_assign_plug:nn {tagsupport/math/display/begin}{noop}
534       \socket_assign_plug:nn {tagsupport/math/display/end}{noop}
535       \socket_assign_plug:nn {tagsupport/math/display/formula/begin}{identity}
536       \socket_assign_plug:nn {tagsupport/math/display/formula/end}{noop}
537     }

```

(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

```

538   \cs_new_protected:Npn \__tag_math_enable:
539     {
540       \socket_assign_plug:nn {tagsupport/math/inline/begin}{MC}
541       \socket_assign_plug:nn {tagsupport/math/inline/end}{MC}
542       \socket_assign_plug:nn {tagsupport/math/inline/formula/begin}{default}
543       \socket_assign_plug:nn {tagsupport/math/inline/formula/end}{default}
544       \socket_assign_plug:nn {tagsupport/math/display/begin}{default}
545       \socket_assign_plug:nn {tagsupport/math/display/end}{default}
546       \socket_assign_plug:nn {tagsupport/math/display/formula/begin}{default}
547       \socket_assign_plug:nn {tagsupport/math/display/formula/end}{default}
548     }

```

(End of definition for __tag_math_enable:.)

At begin document we can activate:

```

549   \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
550   \cs_new_protected:Npn \__math_process:nn #1#2
551     {
552       \legacy_if:nF { measuring@ }
553       {

```

```

554     \tl_if_in:nnTF {#2} { \m@th }
555     { \bool_set_true:N\l__math_fakemath_bool }
556     { \tl_trim_spaces_apply:nN {#2} \__math_process_auxi:nn {#1} }
557   }
558 }
559 \cs_generate_variant:Nn \__math_process:nn { V }
560 \cs_new_protected:Npn \__math_process_auxi:nn #1#2
561 {
562   \tl_gset:Nn \g__math_grabbed_env_tl {#2}
563   \tl_gset:Nn \g__math_grabbed_math_tl {#1}
564   \__math_process_auxii:nn {#2} {#1}
565 }
566 \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

```

567 \cs_new_protected:Npn \math_processor:n #1
568 { \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

`__math_grab_dollar:w`
`__math_grab_dollar:n`

Top-level function to handle grabbing of inline math mode delimited by `$` tokens. We provide two different ways to do that: a token-by-token one that can be used everywhere, and a fast delimited one that does not work anywhere that the end `$` token may be hidden, most obviously in tabulars. The function here is therefore set up as a variable starting point.

```

569 \cs_new_protected:Npn \__math_grab_dollar:w { \__math_grab_dollar_delim:w }

```

After grabbing inline math material, there is again common processing independent of mechanism of collection.

```

570 \cs_new_protected:Npn \__math_grab_dollar:n #1
571 {

```

To avoid further work with entirely empty math segments, we test before doing anything more.

```

572   \tl_if_blank:nF {#1}
573   {
574     \__math_process:nn { math } {#1} % $

```

We do not want math tagging in fakemath or when measuring, We also do not want math tagging if tagging has been suspended.

```

575     \bool_lazy_or:nnTF
576     { \legacy_if_p:n { measuring@ } }
577     { \l__math_fakemath_bool }
578     { #1 $ }
579     {
580       \tag_socket_use:n {math/inline/begin} %end P-MC
581       \tag_socket_use:nn {math/inline/formula/begin}{#1}
582       $
583       \tag_socket_use:n {math/inline/formula/end}
584       \tag_socket_use:n {math/inline/end} % restart P-MC
585     }

```

```

586     }
587   }

```

(End of definition for `_math_grab_dollar:w` and `_math_grab_dollar:n`.)

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

```

588 \cs_new_protected:Npn \_math_grab_dollar_delim:w #1 $ % $
589   { \_math_grab_dollar:n {#1} }

```

(End of definition for `_math_grab_dollar_delim:w`.)

`_math_grab_dollardollar:w` And for the classical T_EX display structure.

```

590 \cs_new_protected:Npn \_math_grab_dollardollar:w % $$
591   #1 $$
592   {
593     \tl_if_blank:nF {#1}
594     {
595       \_math_process:nn { equation* } {#1}
596       \socket_use:n {tagsupport/math/display/begin}
597       \socket_use:nn{tagsupport/math/display/formula/begin}{#1}
598       $$
599     }
600   }

```

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

```

601 \cs_new_protected:Npn \_math_tag_dollardollar_display_end:
602   {
603     % \typeout{== tag dollar\dollar display end}
604     % \ShowTagging{struct-stack}
605     \para_raw_end:

```

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

```

606     \tagpdfpara0n
607     \l_math_tmpa_skip \lastskip
608     \socket_use:n{tagsupport/math/display/formula/end}
609     \penalty \postdisplaypenalty

```

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

```

610 \skip_vertical:n { -\l_math_tmpa_skip * 2 }
611 %
612 \@doendpe           % this has no \end{...} to take care of it
613 }
614

```

(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.

```

615 \cs_new_protected:Npn \_math_grab_inline:w % \ (
616   #1 \ )
617   {
618     \tl_if_blank:nF {#1}
619     {
620       $ #1 $
621     }
622     \bool_set_false:N \l_math_collected_bool
623   }

```

(End of definition for `_math_grab_inline:w`.)

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

```
624 \cs_new_protected:Npn \_math_grab_eqn:w % \[
625   #1 \]
626   {
627   % \typeout{collected? = \bool_if:NTF \l_math_collected_bool {true}{false}}
628   \begin { equation* } #1 \end { equation* }
629   }
```

(End of definition for `_math_grab_eqn:w`.)

5.11 Token-by-token inline grabbing

Grabbing inline math token-by-token is more involved. The mechanism here is essentially a simplified version of that originally seen in `colcell` and refined in `siunitx`. We make use of the fact that in math mode spaces are ignored, so we have to deal with only N-type tokens and groups. Furthermore, there is no need to look inside groups, so the only special cases are a small selection of N-type tokens.

`\l_math_grabbed_tl` For collection of the material piecewise.

```
630 \tl_new:N \l_math_grabbed_tl
```

`\l_math_grab_env_int` Needed to count up the number of nested environments encountered.

```
631 \int_new:N \l_math_grab_env_int
```

`_math_grab_dollar_loop:` The lead-off here establishes a group: we need that as we will have to be careful in the way `\cr` is handled and ensure this is only manipulated whilst grabbing. The main loop is then started.

`_math_grab_loop:`

```
632 \cs_new_protected:Npn \_math_grab_dollar_loop:
633   {
634   \group_begin:
635   \tl_clear:N \l_math_grabbed_tl
636   \_math_grab_loop:
637   }
638 \cs_new_protected:Npn \_math_grab_loop:
639   {
640   \peek_remove_spaces:n
641   {
642   \peek_meaning:NTF \c_group_begin_token
643     { \_math_grab_loop_group:n }
644     { \_math_grab_loop_token:N }
645   }
646   }
```

(End of definition for `_math_grab_dollar_loop:` and `_math_grab_loop:.`)

```

\__math_grab_loop_group:n Handling of grabbed groups is pretty easy.
\__math_grab_loop_store:n
647 \cs_new_protected:Npn \__math_grab_loop_group:n #1
648 { \__math_grab_loop_store:n { {#1} } }
649 \cs_new_protected:Npn \__math_grab_loop_store:n #1
650 {
651   \tl_put_right:Nn \l__math_grabbed_tl {#1}
652   \__math_grab_loop:
653 }

```

(End of definition for `__math_grab_loop_group:n` and `__math_grab_loop_store:n`.)

```

\__math_grab_loop_token:N Filter out the special cases: for performance reasons, use a hash table approach rather
\__math_grab_loop_$: than a loop (cf. collcell). The need to cover \begin_ is that at the start of a cell, TeX will
\__math_grab_loop_\\: expand \begin but the LATEX robust mechanism will mean this yields \begin_. If \begin
\__math_grab_loop_\\begin: were protected, that would not be needed.
\__math_grab_loop_\\begin_:
\__math_grab_loop_\\end:
\__math_grab_loop_\\ignorespaces:
\__math_grab_loop_\\unskip:
\__math_grab_loop_\\textonly@unskip:

```

```

654 \cs_new_protected:Npn \__math_grab_loop_token:N #1
655 {
656   \cs_if_exist_use:cF
657   { \__math_grab_loop_ \token_to_str:N #1 : }
658   { \__math_grab_loop_store:n {#1} }
659 }
660 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N $ : }
661 { \__math_grab_loop_end: }
662 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N \\ : }
663 {
664   \int_compare:nNnTF \l__math_grab_env_int = 0
665   { \__math_grab_loop_newline: }
666   { \__math_grab_loop_store:n { \\ } }
667 }

```

In contrast to `collcell`, nesting is tracked by counting `\begin/\end` pairs: this is needed in case there is a tabular-like construct containing `\\` inside a cell. As a result, the end-of-tabular can be detected without checking the name argument: if `\end` is encountered at nesting level 0, we've hit the end of a cell. In that case, end the row and leave the environment to clean up.

```

668 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N \begin : }
669 {
670   \int_incr:N \l__math_grab_env_int
671   \__math_grab_loop_store:n { \begin }
672 }
673 \cs_new_eq:cc { \__math_grab_loop_ \token_to_str:N \begin \c_space_tl : }
674 { \__math_grab_loop_ \token_to_str:N \begin : }
675 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N \end : }
676 {
677   \int_compare:nNnTF \l__math_grab_env_int = 0
678   {
679     \__math_grab_loop_newline:
680     \end
681   }
682   {
683     \int_decr:N \l__math_grab_env_int
684     \__math_grab_loop_store:n { \end }
685   }
686 }

```

```

687 \tl_map_inline:nn { \ignorespaces \unskip \textonly@unskip }
688 {
689   \cs_new_protected:cpn { __math_grab_loop_ \token_to_str:N #1 : }
690     { \__math_grab_loop: }
691 }

```

(End of definition for `__math_grab_loop_token:N` and others.)

`__math_grab_loop_newline:` To allow collection of tokens in the part of the `\halign` template after `#`, we need `TEX` to see the primitive with the loop token in the right place. That is done by re-defining `\cr` at present. Ideally there would be a socket in the definition of `tabular`, etc., to handle this: there is also the need to examine in interaction with `longtable`, which also redefines `\cr`.

```

692 \cs_new_protected:Npn \__math_grab_loop_newline:
693 {
694   \if_false: { \fi:
695     \cs_set_protected:Npn \cr
696       {
697         \__math_grab_loop:
698         \tex_cr:D
699       }
700     \if_false: } \fi:
701   \l
702 }

```

(End of definition for `__math_grab_loop_newline:.`)

`__math_grab_loop_end:` Clean up and pass on.

```

703 \cs_new_protected:Npn \__math_grab_loop_end:
704 {
705   \exp_args:NNV \group_end:
706   \__math_grab_dollar:n \l__math_grabbed_tl
707 }

```

(End of definition for `__math_grab_loop_end:.`)

5.12 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”

```

708 \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 `lTcmd`.

```

709 \keys_define:nn { __math }
710 {
711   arg-spec .code:n =
712     {
713       \ExpandArgs { c } \DeclareDocumentCommand

```

```

714         { __math_env \l__math_env_name_tl _aux: }
715         {#1}
716         { \__math_env_forward:w }
717     }
718 }

```

`\math_register_env:nn` Set up to capture environment content and make available.
`\math_register_env:n`
`\RegisterMathEnvironment`

```

719 \cs_new_protected:Npn \math_register_env:nn #1#2
720 {
721     \tl_set:Nn \l__math_env_name_tl {#1}
722     \keys_set:nn { __math } {#2}
723     \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
724     \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
725 %
726     \ExpandArgs { nne } \RenewDocumentEnvironment {#1} { b }
727     {
728         \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
729         {
730             \typeout{===>B1}
731         }
732         {
733             \typeout{===>B2}
734             \cs_if_exist:cTF { __math_env #1 _aux: }
735             {
736                 \exp_not:c { __math_env #1 _aux: }
737                 ##1 \exp_not:N \__math_env_end: {#1}
738             }
739             { \exp_not:N \__math_process:nn {#1} {##1} }
740             \exp_not:n { \@kernel@math@registered@begin }
741             \bool_set_true:N \exp_not:N \l__math_collected_bool
742         }
743 %
744         \exp_not:N \tracingall
745         \exp_not:c { __math_env_ #1 _begin: }
746         ##1
747         \exp_not:c { __math_env_ #1 _end: }
748 %
749         \exp_not:N \tracingnone
750     }
751 }
752
753 \cs_new_protected:Npn \math_register_halign_env:nn #1#2
754 {
755     \tl_set:Nn \l__math_env_name_tl {#1}
756     \keys_set:nn { __math } {#2}
757     \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
758     \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
759 %
760     \ExpandArgs { nne } \RenewDocumentEnvironment {#1} { b }
761     {
762         \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
763         {
764             \typeout{===>B1}
765         }

```

```

766     {
767 %       \typeout{===>B2}
768     \cs_if_exist:cTF { __math_env #1 _aux: }
769     {
770       \exp_not:c { __math_env #1 _aux: }
771         ##1 \exp_not:N \_math_env_end: {#1}
772     }
773     { \exp_not:N \_math_process:nn {#1} {##1} }
774     \exp_not:n { \@kernel@math@registered@begin }
775     \bool_set_true:N \exp_not:N \l__math_collected_bool
776   }
777 %   \exp_not:N \tracingall
778   \exp_not:c { __math_env_ #1 _begin: }
779   ##1
780 %   \exp_not:N \tracingnone
781 }
782 {
783   \exp_not:c { __math_env_ #1 _end: }
784 }
785 }

```

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

```

786 \cs_new_protected:Npn \math_register_odd_env:nn #1#2
787 {
788   \tl_set:Nn \l__math_env_name_tl {#1}
789   \keys_set:nn { __math } {#2}
790   \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
791   \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
792 %
793   \ExpandArgs { nnee } \RenewDocumentEnvironment {#1} { b }
794   {
795     \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
796     {
797 %       \typeout{===>B1}
798     }
799     {
800 %       \typeout{===>B2}
801       \cs_if_exist:cTF { __math_env #1 _aux: }
802       {
803         \exp_not:c { __math_env #1 _aux: }
804           ##1 \exp_not:N \_math_env_end: {#1}
805       }
806       { \exp_not:N \_math_process:nn {#1} {##1} }
807       \exp_not:n { \@kernel@math@registered@begin }
808       \bool_set_true:N \exp_not:N \l__math_collected_bool
809     }
810 %   \exp_not:N \tracingall
811   \exp_not:c { __math_env_ #1 _begin: }
812   ##1
813 }
814 {
815   \exp_not:c { __math_env_ #1 _end: }
816 % needed if we don't have $$...$$
817 %   \exp_not:n { \typeout{---> \@kernel@math@registered@end }}

```

```

818     \exp_not:n { \@kernel@math@registered@end }
819   }
820 }
821
822
823 % FMi: compare with block change!
824 %
825 % \DeclareRobustCommand*\begin[1]{%
826 % \UseHook{env/#1/before}%
827 % \ifundefined{#1}%
828 %   {\def\reserved@a{\@latex@error{Environment #1 undefined}\@eha}}%
829 %   {\def\reserved@a{\def\@currenvir{#1}%
830 %     \edef\@currenvline{\on@line}%
831 %     \@execute@begin@hook{#1}%
832 %     \csname #1\endcsname}}%
833 % \ignorefalse
834 % \begingroup
835 % \endpefalse % tmp!!! is it ok to drop this here?
836 % \reserved@a}
837
838
839 \cs_new:Npn \@kernel@math@registered@begin {
840 % \ShowTagging{struct-stack}
841 %\typeout{==>A1}\ShowTagging{struct-stack,mc-current}
842 \mode_if_vertical:TF
843 {
844 %   \legacy_if:nTF { @endpe }
845 %     { \legacy_if_set_false:n { @endpe } }
846 %     { \__block_list_beginpar_vmode: }
847 %
848 %   \typeout{==>~ at:~ \g__tag_struct_tag_tl}
849 %
850 \tag_if_active:T
851 {
852   \exp_args:Noo\str_if_eq:nnF \g__tag_struct_tag_tl { \l__tag_para_main_tag_tl }
853   {
854 %     \typeout{==>A2}
855 %     \__block_beginpar_vmode:
856 %   } % needs correction!
857 }
858 }
859 {
860 %   \typeout{==>A3}
861 %   \__tag_tool_close_P:
862 }
863 \socket_use:nn{tagsupport/math/display/formula/begin/}{}
864 \tagpdfparaOff
865 % \typeout{==>MC1}\ShowTagging{mc-current}
866 }
867
868 \cs_new:Npn \@kernel@math@registered@end {
869 % \typeout{==>MC2}\ShowTagging{mc-current}
870 \para_raw_end:
871 \tagpdfparaOn

```

```

872 \socket_use:n{tagssupport/math/display/formula/end}
873 % \typeout{==>MC3}\ShowTagging{mc-current}
874 \@endpetrue
875 }
876
877 \cs_new_protected:Npn \math_register_env:n #1
878 { \math_register_env:nn {#1} { } }
879
880 \NewDocumentCommand \RegisterMathEnvironment { 0{} m }
881 { \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`

```

882 \cs_new_protected:Npn \_math_env_forward:w #1 \_math_env_end: #2
883 { \_math_process:nn {#2} {#1} }

```

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

5.13 Document commands

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

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

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

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

```

884 \tl_gput_right:Nn \@kernel@before@begindocument
885 {
886   \math_register_env:n { equation }
887   \math_register_env:n { equation* }
888   % at the moment register_env can only do display math
889   % \math_register_env:n { math }
890   \RenewDocumentEnvironment{math} {b}{\$#1$}{}
891   % and this one doesn't work either
892   % \math_register_env:n { displaymath }
893   \RenewDocumentEnvironment{displaymath} {b}{\[#1\]}{}
894 }

```

(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.

```

895 \cs_gset_protected:Npn \( % \)
896 {
897   \bool_if:NTF \l_math_collected_bool
898   {
899     \mode_if_math:TF

```

```

900     { \@badmath }
901     { $ }
902   }
903   {
904     \_math_grab_inline:w
905   }
906 } % \(\
907 \cs_gset_protected:Npn \)
908 {
909   \mode_if_math:TF
910   { $ }
911   { \@badmath }
912 }

```

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

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

```

913 \tl_gput_right:Nn \@kernel@before@begindocument
914 {
915   \cs_gset_protected:Npn \[ % \]
916   {
917     \_math_grab_eqn:w
918   %   \bool_if:NTF \l__math_collected_bool
919   %     { \begin { equation* } }
920   %     { \_math_grab_eqn:w }
921   } % \[
922   \cs_gset_protected:Npn \]
923   {
924     \@badmath
925   %   \bool_if:NTF \l__math_collected_bool
926   %     { \end{ equation* } }
927   %     { \@badmath }
928   }
929 }

```

(End of definition for \[and \]. 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.

```

930 %\cs_gset_protected:Npn \ensuremath #1
931 % {
932 %   \mode_if_math:TF
933 %     {#1}
934 %     {
935 %       \bool_if:NTF \l__math_collected_bool
936 %         { \@ensuredmath {#1} }
937 %         {
938 %           \bool_set_true:N \l__math_collected_bool
939 %           \_math_process:nn { math } {#1}
940 %           \@ensuredmath {#1}
941 %           \bool_set_false:N \l__math_collected_bool
942 %         }
943 %     }
944 % }

```

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

5.14 `\everymath` and `\everydisplay`

The business end for grabbing inline math and “raw” TeX display. Most display math mode is actually handled elsewhere, as we have macro control.

```
945
946 \exp_args:No \tex_everymath:D
947   {
948     \tex_the:D \tex_everymath:D
949     \bool_if:NF \l__math_collected_bool
950     {
951       \bool_set_true:N \l__math_collected_bool
952       \__math_grab_dollar:w
953     }
954   }
955
956 \exp_args:No \tex_everydisplay:D
957   {
958     \tex_the:D \tex_everydisplay:D
959     \iftrue % this may have to be a settable flag!
960     % \typeout{==>~ in~ everydisplay}
```

flipping the `\belowdisplay` values is done so that we get (assumption) a negative skip and not make the page bigger than 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)

```
961     \skip_set:Nn \belowdisplayskip {-\belowdisplayskip}
962     \skip_set:Nn \belowdisplayshortskip {-\belowdisplayshortskip}
963     \int_set:Nn \postdisplaypenalty {10000}
964     \group_insert_after:N \__math_tag_dollardollar_display_end:
965     \fi
966     \bool_if:NF \l__math_collected_bool
967     {
968       \bool_set_true:N \l__math_collected_bool
969       \__math_grab_dollardollar:w
970     }
971   }
```

5.15 Modifying kernel environments

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

```
972 \math_register_env:n { eqnarray }
973 \math_register_env:n { eqnarray* }
```

Tabulars currently contain a `$` that shouldn't trigger math tagging.

```
974 \RequirePackage{array}
975 \tl_if_in:NnT\@tabular{$}
976 {
977   \def\@tabular{%
978     \leavevmode
```

```

979 \UseTaggingSocket{tbl/hmode/begin}%
980 \hbox \bgroup
981 \bool_set_true:N \l__math_collected_bool
982 $
983 \bool_set_false:N \l__math_collected_bool
984 \col@sep\tabcolsep \let\dollarbegin\begin\group
985 \let\dollarend\endgroup

```

A proper switching mechanism is needed: for the present, do directly.

```

986 \cs_set_protected:Npn \__math_grab_dollar:w { \__math_grab_dollar_loop: }
987 \@tabarray}
988 }

```

`__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!

```

989 \cs_new_eq:NN \__math_m@th: \m@th
990 \cs_gset_protected:Npn \m@th
991 {
992   \bool_set_true:N \l__math_collected_bool
993   \__math_m@th:
994 }

```

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

5.16 Disable math grabbing in the `begindocument` hook

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

```

995 \tl_gput_right:Nn\@kernel@before@begindocument
996 {
997   \bool_set_true:N\l__math_collected_bool
998 }
999 \tl_gput_right:Nn\@kernel@after@begindocument
1000 {
1001   \bool_set_false:N\l__math_collected_bool
1002 }

```

5.17 Modifying `amsmath`

`__math_amsmath_align@:nn` 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`.

```

1003
1004
1005
1006 \tl_gput_right:Nn \@kernel@before@begindocument {
1007   %
1008   \renewenvironment{gather*}{%
1009     \start@gather\st@rredtrue
1010   }
1011   {%

```

```

1012 % this redirection doesn't work if we alter "gather"!
1013 % \endgather
1014 % so replace it with its real meaning
1015 \math@cr \black@totwidth@ \egroup
1016 $$\ignorespacesafterend
1017 }

1018 \def\common@align@ending {
1019 \math@cr \black@totwidth@
1020 \egroup
1021 \ifingather@
1022 \restorealignstate@
1023 \egroup
1024 \nonumber
1025 \ifnum0='{\fi\iffalse}\fi
1026 \else
1027 $$%
1028 \fi
1029 \ignorespacesafterend
1030 }

1031 \renewenvironment{alignat}{%
1032 \start@align\z@\st@rredfalse
1033 }{%
1034 \common@align@ending
1035 }

1036 \renewenvironment{alignat*}{%
1037 \start@align\z@\st@rredtrue
1038 }{%
1039 \common@align@ending
1040 }

1041 \renewenvironment{xalignat}{%
1042 \start@align\@ne\st@rredfalse
1043 }{%
1044 \common@align@ending
1045 }

1046 \renewenvironment{xalignat*}{%
1047 \start@align\@ne\st@rredtrue
1048 }{%
1049 \common@align@ending
1050 }

1051 \renewenvironment{xxalignat}{%
1052 \start@align\tw@\st@rredtrue
1053 }{%
1054 \common@align@ending
1055 }

1056 \renewenvironment{align}{%
1057 \start@align\@ne\st@rredfalse\m@ne
1058 }{%
1059 \common@align@ending
1060 }

1061 \renewenvironment{align*}{%
1062 \start@align\@ne\st@rredtrue\m@ne
1063 }{%
1064 \common@align@ending
1065 }

```

```

1066 \renewenvironment{flalign}{%
1067   \start@align\tw@\st@rredfalse\m@ne
1068 }{%
1069   \common@align@ending
1070 }
1071 \renewenvironment{flalign*}{%
1072   \start@align\tw@\st@rredtrue\m@ne
1073 }{%
1074   \common@align@ending
1075 }
1076 %
1077 \renewenvironment{multline*}{\start@multline\st@rredtrue}
1078 {%
1079   \iftagsleft@ \@xp\lendmultline@ \else \@xp\rendmultline@ \fi
1080   \ignorespacesafterend
1081 }

```

Also for false?

```

1082 \def\measuring@true{\let\ifmeasuring@\iftrue\tag_stop:}
1083 %
1084 \math_register_halign_env:nn {align}{}
1085 \math_register_halign_env:nn {align*}{}
1086 \math_register_halign_env:nn {alignat}{}
1087 \math_register_halign_env:nn {alignat*}{}
1088 \math_register_halign_env:nn {flalign}{}
1089 \math_register_halign_env:nn {flalign*}{}
1090 \math_register_halign_env:nn {gather}{}
1091 \math_register_halign_env:nn {gather*}{}
1092 \math_register_halign_env:nn {multline}{}
1093 \math_register_halign_env:nn {multline*}{}
1094 \math_register_halign_env:nn {xalignat}{}
1095 \math_register_halign_env:nn {xalignat*}{}
1096 \math_register_halign_env:nn {xxalignat}{}
1097 %
1098 \@namedef{maketag @ @ @} #1{%
1099 %   \typeout{-->maketag @ @ @}
1100   \ifmeasuring@
1101     \hbox{\m@th\normalfont#1}%
1102   \else
1103     \tagmccend \tagstructbegin{tag=Lbl}%
1104     \tagmcbegin{tag=Lbl}%
1105     \hbox{\m@th\normalfont#1}%
1106     \tagmccend \tagstructend \tagmcbegin{}%
1107   \fi
1108 }
1109 \@namedef{math@cr @ @ @ gather}{%
1110   \ifst@rred\nonumber\fi
1111   &\relax
1112   \make@display@tag
1113 %
1114 \maybestartnewformulatag
1115 %
1116 \ifst@rred\else\global\@eqnswtrue\fi
1117 \global\advance\row@\@ne

```

```

1118     \cr
1119 }

1120 \@namedef{math@cr @ @ @ align}{%
1121 \ifst@rred\nonumber\fi
1122 \if@eqnsw \global\tag@true \fi
1123 \global\advance\row@\@ne
1124 \add@amps\maxfields@
1125 \omit
1126 \kern-\alignsep@
1127 \iftag@
1128 \setboxz@h{\@lign\strut@{\make@display@tag}}%
1129 \place@tag
1130 \fi
1131 %
1132 \maybestartnewformulatag
1133 %
1134 \ifst@rred\else\global\@eqnswtrue\fi
1135 \global\lineht@\z@
1136 \cr
1137 }

1138 \def\restore@math@cr{\@namedef{math@cr @ @ @}{
1139 %
1140 \maybestartnewformulatag
1141 %
1142 \cr}}
1143 \restore@math@cr
1144 }

```

(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.

```

1145 \cs_new:Npn \_math_split_at_nl:NN #1#2 {
1146 \tl_set:Nf \l_math_tmpa_tl {
1147 \exp_after:wN \_math_split_at_nl_first:w #1 \ \ \q_nil \ \s_stop }
1148 \exp_after:wN \_math_split_at_nl_aux:nnNN \l_math_tmpa_tl #1 #2
1149 }

```

and the auxiliary commands

```

1150 \cs_new:Npn \_math_split_at_nl_first:w #1 \ \ #2 \ \ #3 \s_stop
1151 {
1152 \quark_if_nil:nTF {#2}
1153 { {#1} { } }
1154 {
1155 \_math_split_chk_if_begin:ww #1 \begin \q_nil \s_mark
1156 #2 \ \ #3 \s_stop
1157 }
1158 }
1159
1160 \cs_new_protected:Npn \_math_split_at_nl_aux:nnNN #1 #2 #3 #4
1161 {
1162 \tl_gset:Nn #4 {#1}
1163 \tl_gset:Nn #3 {#2}

```

```

1164 }
1165
1166 \cs_new:Npn \__math_split_chk_if_begin:ww
1167   #1 \begin #2 #3 \s_mark #4 \ \q_nil \ \s_stop
1168 {
1169   \quark_if_nil:nTF {#2}
1170     { {#1} {#4} }
1171     {
1172       \exp_after:wN \__math_split_collect_one_end:w
1173       \__math_split_cleanup_begin_q_nil:w #1 \begin{#2} #3 \ \ #4 \s_stop
1174       { } { 1 }
1175     }
1176 }
1177
1178 \cs_new:Npn \__math_split_cleanup_begin_q_nil:w #1 \begin \q_nil {#1}
1179
1180 \cs_new:Npn \__math_split_collect_one_end:w #1 \end #2 #3 \s_stop #4 #5
1181 {
1182   \exp_args:Nf \__math_split_check_count_begins:n
1183     { \__math_split_count_begins:n { #4 #1 } } {#5}
1184     { #4 #1 \end{#2} } {#3}
1185 }
1186 \cs_new:Npn \__math_split_count_begins:n #1
1187 { \int_eval:n { 0 \__math_split_count_begins:w #1 \begin \q_nil } }
1188
1189 \cs_new:Npn \__math_split_count_begins:w #1 \begin #2
1190 { \quark_if_nil:nF {#2} { +1 \__math_split_count_begins:w } }
1191
1192 \cs_new:Npn \__math_split_check_count_begins:n #1 #2 #3 #4
1193 {
1194   \int_compare:nNnTF {#1} = {#2}
1195     {
1196       \exp_last_unbraced:Nf \__math_split_final_cleanup:nn
1197       { \__math_split:n { \__math_split_guard:n {#3} #4 } }
1198     }
1199     {
1200       \exp_args:No \use_i_i:n
1201       { \exp_after:wN { \int_value:w \int_eval:n { #2 + 1 } } }
1202       { \__math_split_collect_one_end:w #4 \s_stop {#3} }
1203     }
1204 }
1205 \cs_new:Npn \__math_split_final_cleanup:nn #1 #2
1206 {
1207   \exp:w \__math_split_final_cleanup:w #1
1208   \__math_split_guard:n \q_nil \s_mark { }
1209   {#2}
1210 }
1211 \cs_new:Npn \__math_split_final_cleanup:w #1 \__math_split_guard:n #2 #3 \s_mark #4
1212 {
1213   \quark_if_nil:nTF {#2}
1214     { \exp_end: { #4 #1 } }
1215     { \__math_split_final_cleanup:w #3 \s_mark { #4 #1 #2 } }
1216 }
1217

```

```

1218 \cs_new:Npn \__math_split:n #1 {
1219     \__math_split_at_nl_first:w #1 \ \q_nil \ \s_stop }
1220
1221 % this looks unused.
1222 %\NewDocumentCommand \splitnl { mm +m }
1223 % {
1224 %     \tl_set:Nf \l__math_tmpa_tl { \split:n {#3} }
1225 %     \show \l__math_tmpa_tl
1226 %     \exp_after:wN \__splitnl_aux:nnNN \l__math_tmpa_tl #1 #2
1227 % }

```

(End of definition for __math_split_at_nl:NN.)

\maybestartnewformulatag

```

1228
1229 \newif\if@subformulas
1230 \tl_new:N \result
1231
1232 \cs_new_protected:Npn\grabaformulapartandstart {
1233     \__math_split_at_nl:NN \g__math_grabbed_math_tl \result
1234     \typeout{====>first-result=\meaning\result}
1235     \typeout{====>first-tmpmathcontent=\meaning\g__math_grabbed_math_tl}
1236     \tl_if_empty:NTF \g__math_grabbed_math_tl
1237     {
1238         \typeout{====>formula~ has~ no~ subparts}
1239         \global\@subformulasfalse
1240     }
1241     {
1242         \typeout{====>formula~ has~ subparts}
1243         \global\@subformulastrue
1244         \edef\resulttitle{\g__math_grabbed_env_tl\space (part)}
1245         \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

```

1246 %         alt=\result,
1247 %         alt = subformula,
1248 %         title-o=\resulttitle
1249 %     }
1250 % }
1251 % \tagmcbegin{}
1252 % }
1253
1254 \cs_new_protected:Npn\grabaformulapartandmayberestart {
1255     \__math_split_at_nl:NN \g__math_grabbed_math_tl \result
1256     \typeout{====>result=\meaning\result}
1257     \typeout{====>tmpmathcontent=\meaning\g__math_grabbed_math_tl}
1258 %     \tl_if_empty:NTF \g__math_grabbed_math_tl
1259 %     {
1260 %         \typeout{====>tmpmathcontent=empty}
1261 %     }
1262 %     {
1263 %         \typeout{====>tmpmathcontent=not-empty}
1264 %         \edef\resulttitle{\g__math_grabbed_env_tl\space (part)}
1265 %         \tagstructbegin{tag=Formula,

```

```

1266         alt=\result,
1267         title-o=\resulttitle
1268     }
1269 % }
1270 \tagmcbegin{}
1271 }

```

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

```

1272 \def\maybestartnewformulatag {
1273 \if@subformulas
1274 \ifmeasuring@\else
1275 %
1276 \tl_if_empty:NF \g__math_grabbed_math_tl
1277 {
1278     \tagmcbegin
1279     \tagmcbegin
1280     \grabaformulapartandmayberestart
1281 }
1282 \fi
1283 \fi
1284 }

```

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

```

1285 %\AddToHook{package/breqn/after}{
1286 % \typeout{===>~ in~ hook}
1287 % \math_register_halign_env:nn {dmath}{}
1288 % \math_register_halign_env:nn {dgroup*}{}
1289 %}
1290 \ExplSyntaxOff
1291 <@@=
1292 </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>	72, 228
<code>\%</code>	73
<code>\(</code>	615
<code>\(</code>	<u>895</u>
<code>\)</code>	616
<code>\)</code>	<u>895</u>
<code>@@</code> commands:	
<code>\l_@@_collected_bool</code>	3, 7
<code>\l_@@_fakemath_bool</code>	8
<code>\[</code>	624, 893
<code>\[</code>	9, 26, 32, 33, <u>913</u>
<code>\backslash</code>	134, 135, 136, 662, 666, 701, 1147, 1150, 1156, 1167, 1173, 1219
<code>\{</code>	70
<code>\}</code>	71
<code>\]</code>	625, 893
<code>\]</code>	9, 26, 32, 33, <u>913</u>
A	
<code>actual+source (plug)</code>	<u>381</u>
<code>\AddToHook</code> 7, 9, 121, 167, 209, 250, 267, 1285	
<code>\advance</code>	1117, 1123
<code>\aftergroup</code>	25
<code>alt+source (plug)</code>	<u>396</u>

<code>\AssignSocketPlug</code>	142, 143, 144, 145, 149, 207, 208, 246, 247, 248, 249	<code>\cs_if_exist:N</code>	484
<code>\AtBeginDocument</code>	549	<code>\cs_if_exist_use:N</code>	656
B			
<code>\begin</code>	27, 357, 375, 628, 668, 671, 673, 674, 825, 919, 1155, 1167, 1173, 1178, 1187, 1189	<code>\cs_new:Npn</code>	153, 839, 868, 1145, 1150, 1166, 1178, 1180, 1186, 1189, 1192, 1205, 1211, 1218
<code>\beginngroup</code>	69, 834, 984	<code>\cs_new_eq:NN</code>	673, 989
<code>\belowdisplay</code>	34	<code>\cs_new_protected:Npn</code>	22, 53, 68, 76, 139, 220, 271, 527, 538, 550, 560, 566, 567, 569, 570, 588, 590, 601, 615, 624, 632, 638, 647, 649, 654, 660, 662, 668, 675, 689, 692, 703, 719, 753, 786, 877, 882, 1160, 1232, 1254
<code>\belowdisplayskip</code>	962	<code>\cs_set_eq:NN</code>	229
<code>\belowdisplayskip</code>	34, 961	<code>\cs_set_protected:Npn</code>	568, 695, 986
<code>\bgroup</code>	980	<code>\csname</code>	832
block internal commands:		D	
<code>__block_beginpar_vmode:</code>	855	<code>\DeclareDocumentCommand</code>	713
<code>__block_list_beginpar_vmode:</code>	846	<code>\DeclareMathEnvironment</code>	8
bool commands:		<code>\DeclareRobustCommand</code>	825
<code>\bool_gset_true:N</code>	44, 141, 195, 236	<code>\def</code>	828, 829, 977, 1018, 1082, 1138, 1272
<code>\bool_if:N</code>	62, 123, 243, 418, 455, 460, 474, 482, 627, 728, 762, 795, 897, 918, 925, 935, 949, 966	default (plug)	305, 327, 336, 414, 499
<code>\bool_lazy_or:mnTF</code>	575	<code>\displaylines</code>	8
<code>\bool_new:N</code>	10, 11, 37, 38, 39, 40, 41, 42, 43	E	
<code>\bool_set_false:N</code>	622, 941, 983, 1001	<code>\edef</code>	830, 1244, 1264
<code>\bool_set_true:N</code>	555, 741, 775, 808, 938, 951, 968, 981, 992, 997	<code>\egroup</code>	1015, 1020, 1023
bool internal commands:		<code>\else</code>	1026, 1079, 1102, 1116, 1134, 1274
<code>\l_math_collected_bool</code>	9, 10, 622, 627, 728, 741, 762, 775, 795, 808, 897, 918, 925, 935, 938, 941, 949, 951, 966, 968, 981, 983, 992, 997, 1001	<code>\end</code>	27, 361, 377, 612, 628, 675, 680, 684, 926, 1180, 1184
<code>\l_math_fakemath_bool</code>	9, 11, 555, 577	end internal commands:	
box commands:		<code>__math_tag_dollardollar_</code>	
<code>\box_new:N</code>	219	<code>display_end</code>	34
box internal commands:		<code>\endcsname</code>	832
<code>\l_math_tmpa_box</code>	219, 222	<code>\endequation</code>	884
<code>\boxed</code>	7	<code>\endequation*</code>	884
C			
char commands:		<code>\endgather</code>	1013
<code>\char_set_catcode_other:N</code>	70, 71, 72, 73, 228	<code>\endgroup</code>	77, 985
clist commands:		<code>\ensuremath</code>	8, 930
<code>\clist_map_inline:Nn</code>	230	<code>\eqalign</code>	8
<code>\clist_new:N</code>	50	<code>\equation</code>	884
<code>\clist_put_right:Nn</code>	51	<code>\equation*</code>	884
<code>\cr</code>	695, 1118, 1136, 1142	<code>\everydisplay</code>	34
cs commands:		<code>\everymath</code>	3, 33, 34
<code>\cs_generate_variant:Nn</code>	280, 437, 559	exp commands:	
<code>\cs_gset_eq:NN</code>	723, 724, 757, 758, 790, 791	<code>\exp:w</code>	1207
<code>\cs_gset_protected:Npn</code>	895, 907, 915, 922, 930, 990	<code>\exp_after:wN</code>	1147, 1148, 1172, 1201, 1226
		<code>\exp_args:Nf</code>	1182
		<code>\exp_args:NNV</code>	705
		<code>\exp_args:No</code>	946, 956, 1200
		<code>\exp_args:Noo</code>	852

<code>\legacy_if_set_false:n</code>	845	<code>__math_grab_loop_begin:</code>	654
<code>\let</code>	984, 985, 1082	<code>__math_grab_loop_end:</code>	654
<code>\ltxlabmathdate</code>	4	<code>__math_grab_loop_ignorespaces:</code>	654
<code>\ltxlabmathversion</code>	5	<code>__math_grab_loop_textonly@unskip:</code>	654
luamml commands:		654
<code>\luamml_flag_process:</code>	166	<code>__math_grab_loop_unskip:</code>	654
luamml internal commands:		<code>__math_grab_loop_end:</code> .	661, 703, 703
<code>\l__luamml_pretty_int</code>	146	<code>__math_grab_loop_group:n</code>	643, 647, 647
<code>__luamml_register_output_hook:N</code>	165	665, 679, 692, 692
		<code>__math_grab_loop_newline:</code>	665, 679, 692, 692
		647, 648, 649, 658, 666, 671, 684
		<code>__math_grab_loop_store:n</code>	644, 654, 654
		126, 139
		<code>__math_grab_loop_token:N</code>	153, 165
		989, 989, 993
		<code>__math_grab_loop_group:n</code>	550, 550,
		559, 574, 595, 739, 773, 806, 883, 939
		<code>__math_grab_loop_newline:</code>	550, 556, 560
		550, 564, 566, 568
		<code>__math_grab_loop_store:n</code>	1197, 1218
		1145, 1145, 1233, 1255
		<code>__math_grab_loop_token:N</code>	1148, 1160
		1147, 1150, 1219
		<code>__math_grab_loop_group:n</code>	1182, 1192
		1155, 1166
		<code>__math_grab_loop_newline:</code>	1173, 1178
		1172, 1180, 1202
		<code>__math_grab_loop_store:n</code>	1183, 1186
		1187, 1189, 1190
		<code>__math_grab_loop_token:N</code>	1196, 1205
		1207, 1211, 1215
		<code>__math_grab_loop_group:n</code>	1197, 1208, 1211
		601, 964
		<code>__math_grab_loop_newline:</code>	271, 271, 280
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		
		<code>__math_grab_loop_group:n</code>	
		
		<code>__math_grab_loop_newline:</code>	
		
		<code>__math_grab_loop_store:n</code>	
		
		<code>__math_grab_loop_token:N</code>	
		

tagsupport/math/display/end	...	323	\tag_struct_end:	29 , 433 , 495
tagsupport/math/display/formula/begin	323	tag internal commands:		
tagsupport/math/display/formula/end	323	_tag_check_para_end_show:nn	...	28
tagsupport/math/end	526	_tag_gincr_para_end_int:	27
tagsupport/math/inline/begin	..	293	_l_tag_math_alt_bool	10 , 42 , 284 , 455	
tagsupport/math/inline/end	293	_tag_math_disable:	527 , 527
tagsupport/math/inline/formula/begin	293	_tag_math_enable:	...	538 , 538 , 549
tagsupport/math/inline/formula/end	293	_g_tag_math_luamml_bool	
tagsupport/math/mathml/write	..	180	10 , 43 , 44 , 123 , 177	
tagsupport/math/mathml/write/prepare	98	_g_tag_math_mathml_AF_bool	
tagsupport/math/struct/begin	..	412	10 , 40 , 141 , 195 , 236 , 243	
tagsupport/math/struct/end	412	_l_tag_math_mathml_AF_bool	
tagsupport/math/substruct/begin	497		10 , 39 , 288 , 460 , 482	
tagsupport/math/substruct/end	..	497	_l_tag_math_mathml_files_clist	..	
\space	4 , 5 , 1244 , 1264	11 , 50 , 51 , 230 , 283	
split commands:			_l_tag_math_mathml_pane_bool	...	
\split:n	1224	10 , 41 , 62 , 285	
\splittl	1222	_l_tag_math_texsource_AF_bool	..	
splitnl internal commands:			10 , 37 , 290 , 418 , 474	
_splittl_aux:nnNN	1226	_l_tag_math_texsource_pane_bool	10 , 38 , 287
str commands:			_l_tag_para_main_tag_tl	852
_c_backslash_str	108	_g_tag_struct_tag_tl	848 , 852
_str_if_eq:nnTF	852	_tag_tool_close_P:	22 , 22 , 330 , 861	
_str_mdfive_hash:n	437 , 446	\tagmcbegin	1104 , 1106 , 1251 , 1270
_str_new:N	16	\tagmcend	507 , 1103 , 1106 , 1278
_str_replace_all:Nnn	102 , 103	\tagpdfparaOff	864
_str_set:Nn	101	\tagpdfparaOn	606 , 871
str internal commands:			\tagpdfsetup	33 , 137
_l_math_tmpa_str	9 , 16 , 101 , 102 , 103 , 110	\tagstructbegin	1103 , 1245 , 1265
\symletters	148	\tagstructend	509 , 1106 , 1279
\symsymbols	147	tagsupport/math/content (socket)	...	380
sys commands:			tagsupport/math/display/begin (socket)	323
_sys_if_engine luatex:TF	116	tagsupport/math/display/end (socket)	323
_c_sys_jobname_str	52 , 151 , 201	tagsupport/math/display/formula/begin (socket)	323
			tagsupport/math/display/formula/end (socket)	323
			tagsupport/math/end (socket)	526
			tagsupport/math/inline/begin (socket)	293
			tagsupport/math/inline/end (socket)	..	293
			tagsupport/math/inline/formula/begin (socket)	293
			tagsupport/math/inline/formula/end (socket)	293
			tagsupport/math/mathml/write (socket)	180
			tagsupport/math/mathml/write/prepare (socket)	98
			tagsupport/math/struct/begin (socket)	412
			tagsupport/math/struct/end (socket)	..	412
			tagsupport/math/substruct/begin (socket)	497
			tagsupport/math/substruct/end (socket)	497
			tagsupport/math/substruct/end (socket)	497

T

\tabcolsep	984
tag commands:		
\tag_if_active:TF	24 , 549 , 850
\tag_mc_begin:n	520
\tag_mc_begin_pop:n	304
\tag_mc_end:	26 , 525
\tag_mc_end_push:	26 , 300
\tag_socket_use:n	580 , 583 , 584
\tag_socket_use:nn	581
\tag_start:	278
\tag_stop:	278 , 1082
\tag_struct_begin:n	4 , 421 , 477

tagsupport/math/substruct/end (socket)		<code>\lineht@</code>	1135
test-mathml (plug)	497	<code>\m@ne</code>	1057, 1062, 1067, 1072
TeX and L ^A T _E X 2 _ε commands:	437	<code>\m@th</code>	7–9, 25, 35, 554, 989, 1101, 1105
<code>\@badmath</code>	900, 911, 924, 927	<code>\make@display@tag</code>	1112, 1128
<code>\@currenvir</code>	829	<code>\math@cr</code>	1015, 1019
<code>\@currenvline</code>	830	<code>\maxfields@</code>	1124
<code>\@doendpe</code>	612	<code>\measuring@true</code>	1082
<code>\@eha</code>	828	<code>\multline@</code>	1003
<code>\@endpefalse</code>	835	<code>\on@line</code>	830
<code>\@endpetrue</code>	874	<code>\place@tag</code>	1129
<code>\@ensuredmath</code>	936, 940	<code>\rendmultline@</code>	1079
<code>\@eqnswtrue</code>	1116, 1134	<code>\reserved@a</code>	828, 829, 836
<code>\@execute@begin@hook</code>	831	<code>\restore@math@cr</code>	1138, 1143
<code>\@ifpackageloaded</code>	125	<code>\restorealignstate@</code>	1022
<code>\@ifundefined</code>	827	<code>\row@</code>	1117, 1123
<code>\@ignorefalse</code>	833	<code>\setboxz@h</code>	1128
<code>\@iiiparbox</code>	7	<code>\split@tag</code>	35
<code>\@kernel@after@begindocument</code>	999	<code>\st@rredfalse</code>	1032, 1042, 1057, 1067
<code>\@kernel@before@begindocument</code>	884, 913, 995, 1006	<code>\st@rredtrue</code>	1009, 1037, 1047, 1052, 1062, 1072, 1077
<code>\@kernel@math@registered@begin</code>	740, 774, 807, 839	<code>\start@align</code>	1032, 1037, 1042, 1047, 1052, 1057, 1062, 1067, 1072
<code>\@kernel@math@registered@end</code>	818, 868	<code>\start@gather</code>	1009
<code>\@latex@error</code>	828	<code>\start@multline</code>	1077
<code>\@lign</code>	1128	<code>\strut@</code>	1128
<code>\@namedef</code>	1098, 1109, 1120, 1138	<code>\tag@true</code>	1122
<code>\@ne</code>	1042, 1047, 1057, 1062, 1117, 1123	<code>\textonly@unskip</code>	687
<code>\@subformulasfalse</code>	1239	<code>\totwidth@</code>	1015, 1019
<code>\@subformulastrue</code>	1243	<code>\tw@</code>	1052, 1067, 1072
<code>\@tabarray</code>	987	<code>\z@</code>	1032, 1037, 1135
<code>\@tabular</code>	975, 977	tex commands:	
<code>\@xp</code>	1079	<code>\tex_cr:D</code>	698
<code>\add@amps</code>	1124	<code>\tex_everydisplay:D</code>	956, 958
<code>\align@</code>	1003	<code>\tex_everymath:D</code>	946, 948
<code>\alignsep@</code>	1126	<code>\tex_the:D</code>	948, 958
<code>\black@</code>	1015, 1019	<code>\text</code>	16
<code>\col@sep</code>	984	tl commands:	
<code>\common@align@ending</code>	1018, 1034, 1039, 1044, 1049, 1054, 1059, 1064, 1069, 1074	<code>\c_space_tl</code>	108, 254, 256, 258, 260, 262, 358, 360, 362, 673
<code>\cr</code>	26, 28	<code>\tl_clear:N</code>	420, 476, 635
<code>\d@llarbegin</code>	984	<code>\tl_const:Nn</code>	80, 88, 94, 365
<code>\d@llarend</code>	985	<code>\tl_gput_right:Nn</code>	884, 913, 995, 999, 1006
<code>\gather@</code>	1003	<code>\tl_gset:Nn</code>	562, 563, 1162, 1163
<code>\halign</code>	28	<code>\tl_gset_eq:NN</code>	65
<code>\if@eqnsw</code>	1122	<code>\tl_if_blank:nTF</code>	572, 593, 618
<code>\if@subformulas</code>	508, 1229, 1273	<code>\tl_if_empty:NTF</code>	155, 1236, 1258, 1276
<code>\ifingather@</code>	1021	<code>\tl_if_eq:NNTF</code>	368
<code>\ifmeasuring@</code>	3, 8, 1082, 1100, 1274	<code>\tl_if_eq:NnTF</code>	448
<code>\ifst@rred</code>	1110, 1116, 1121, 1134	<code>\tl_if_exist:NTF</code>	57, 196, 457
<code>\iftag@</code>	1127	<code>\tl_if_in:NnTF</code>	975
<code>\iftagsleft@</code>	1079	<code>\tl_if_in:nnTF</code>	554
<code>\lendmultline@</code>	1079	<code>\tl_map_inline:nn</code>	687

