

The `latex-lab-math` code*

Frank Mittelbach, Joseph Wright, L^AT_EX Project

v0.6h 2024-10-12

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 <code>luamml</code>	6
	3.5 Options	6
4	Known current bugs, etc.	6
	4.1 Capture/grabbing problems	7
	4.2 Fake math	7
	4.3 Processor	8
	4.4 Other problems	8
	4.5 Other ToDos	8

*

5	The Implementation	9
5.1	File declaration	9
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	18
5.7	Tagging options	19
5.8	Sockets	19
5.8.1	Main inline math sockets	19
5.8.2	Main display math sockets	20
5.8.3	Internal sockets	21
5.9	Interface commands	26
5.10	Content grabbing	27
5.11	Token-by-token inline grabbing	29
5.12	Marking math environments	31
5.13	Document commands	35
5.14	<code>\everymath</code> and <code>\everydisplay</code>	37
5.15	Modifying kernel environments	37
5.16	Disable math grabbing in the <code>begindocument</code> hook	38
5.17	Modifying <code>amsmath</code>	38
	Index	44

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 `bbox` 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: $\llbracket \dots \rrbracket$ 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 $\langle env \rangle$ 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 $\langle options \rangle$ 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 $\langle tokens \rangle$, 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 $\langle env \rangle$ 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 $\langle options \rangle$ 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. It is not *used* as a tagging socket as the math argument should not be lost without tagging, so 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 found 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`. If `unicode-math` is not used, the generation of the file can be forced with `math/mathml/luamml=true` or `math/mathml/luamml` but be aware that it is then probable that various symbols are mapped to the wrong unicode code points. 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.

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

4.1 Capture/grabbing problems

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

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

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

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

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

4.2 Fake math

In a number of places in L^AT_EX math commands (mainly $\$$) is used only for technical reason, e.g. to access a math font, to setup a symbol or to use `\vcenter`.

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

- After grabbing math content the code checks if the content contains the token `\m@th` and if yes it doesn't call the processor before reinserting the content and perhaps adding tagging code. This method requires that the math can be grabbed (e.g. that the end dollar is visible) and that the `\m@th` is visible. It applies for example in `\@iiiparbox` where the code from `\vcenter` to `\m@th` is grabbed and put back. It does not work for example for `tabular` where the dollars and the `\m@th` token are spread around over three commands. `tabular` needs therefore manual intervention. A look in the list of usages (in `usage-of-m@th.md`) justifies this approach. All usages are either not math at all, or related to small elements that probably shouldn't be grabbed and processed on their own.
- `\m@th` is redefined so that it sets the boolean `\l_@@_collected_bool` to true. If `\m@th` is used inside math that has been grabbed this doesn't change much as the boolean is set by the grabbing anyway. For usages outside math the benefit is not so clear: The setting avoids that in L^AT_EX_ε 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 `ltxcmd` [commands] should not be copied, . . .
 - Package authors should be able to manually set up math environments with a public boolean.

4.5 Other ToDos

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

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

5 The Implementation

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

5.1 File declaration

```
3 \ProvidesFile{latex-lab-math.ltx}
4     [\ltlabmathdate\space
5     v\ltlabmathversion\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

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

```

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:mn{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_tl

```

The variable `\g__tag_math_luamml_tl` is initially 0 and the user key can set it to -1 or 1. This allows to distinguish the unset case from a value set by the user.

```

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 \tl_new:N\g__tag_math_luamml_tl
44 \tl_gset:Nn\g__tag_math_luamml_tl {0}

```

```

\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 }
```

Temporary (!) fixes for endarray

```
121   \AddToHook{package/array/after}
122   {
123     \cs_set:Npn \endarray
124     {
125       \tbl_crcr:n{endarray}
126       \__luamml_array_save_array:
127       \egroup
128       \UseTaggingSocket{tbl/finalize}
129       \tbl_restore_outer_cell_data:
130       \egroup
131       \mode_if_math:T { \__luamml_array_finalize_array: }
132       \@arrayright
133       \gdef \@preamble {}
134     }
135     \cs_set:Npn \@classz
136     {
137       \@classx
138       \@tempcnta \count@
139       \prepnext@tok
140       \@addtopreamble {
141         \ifcase \@chnum
```

```

142     \hfil
143     \hskip 1sp
144     \d@llarbegin
145     \cs_if_eq:NNTF \d@llarbegin \begingroup {
146         \insert@column
147         \d@llarend
148     } {
149         \__luamml_array_init_col:
150         \insert@column
151         \luamml_flag_save:nn {} {mtd}
152         \d@llarend
153         \__luamml_array_finalize_col:w 0~
154     }
155     \do@row@strut
156     \hfil
157 \or
158     \hskip 1sp
159     \d@llarbegin
160     \cs_if_eq:NNTF \d@llarbegin \begingroup {
161         \insert@column
162         \d@llarend
163     } {
164         \__luamml_array_init_col:
165         \insert@column
166         \luamml_flag_save:nn {} {mtd}
167         \d@llarend
168         \__luamml_array_finalize_col:w 1~
169     }
170     \do@row@strut
171     \hfil
172 \or
173     \hfil
174     \hskip 1sp
175     \d@llarbegin
176     \cs_if_eq:NNTF \d@llarbegin \begingroup {
177         \insert@column
178         \d@llarend
179     } {
180         \__luamml_array_init_col:
181         \insert@column
182         \luamml_flag_save:nn {} {mtd}
183         \d@llarend
184         \__luamml_array_finalize_col:w 2~
185     }
186     \do@row@strut
187 \or
188     \setbox \ar@mcellbox \vbox \@startpbox { \@nextchar }
189     \insert@pcolumn
190     \endpbox
191     \ar@align@mcell
192     \do@row@strut
193 \or
194     \vtop \@startpbox { \@nextchar }
195     \insert@pcolumn

```

```

196         \@endpbox
197         \do@row@strut
198     \or
199     \vbox \@startpbox { \@nextchar }
200     \insert@pcolumn
201     \@endpbox
202     \do@row@strut
203     \fi
204 }
205 \prepnext@tok
206 }
207 }
208 \AddToHook{begindocument/end}
209 {
210     \str_case:on \g__tag_math_luamml_tl
211     {
212         { 1 } {
213             \__math_luamml_activate_write:
214             \msg_note:nnnn { tag }
215             { luamml-status }{ enabled }{ create }
216         }
217         {-1 } {
218             \msg_note:nnnn { tag }
219             { luamml-status }{ disabled }{ not~create }
220         }
221         { 0 }
222         {
223             \@ifpackageloaded { unicode-math }
224             {
225                 \__math_luamml_activate_write:
226                 \msg_note:nnnn { tag }
227                 { luamml-status }{ enabled }{ create }
228             }
229             { \msg_warning:nn { tag }{ unicode-math-missing } }
230         }
231     }
232 }
233 }
234 }
235 \msg_new:nnn { tag }{ luamml-status }
236 {
237     luamml~has~been~#1~and~will~#2~an~MathML~file.
238 }
239
240 \msg_new:nnn { tag }{ unicode-math-missing }
241 {
242     The~package~unicode-math~is~missing\\
243     luamml~will~not~create~an~MathML~file.\\
244     To~avoid~this~warning~load~unicode-math~\\
245     or~disable~luamml~with~\\
246     \tl_to_str:n{\tagpdfsetup{math/mathml/luamml=false}}\\
247     or~force~luamml~with~\\
248     \tl_to_str:n{\tagpdfsetup{math/mathml/luamml=true}}
249 }

```

```

250 \cs_new_protected:Npn \__math_luamml_activate_write:
251 {

```

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

```

252   \bool_gset_true:N\g__tag_math_mathml_AF_bool
253   \AssignSocketPlug{tagsupport/math/struct/begin}{test-mathml}
254   \AssignSocketPlug{tagsupport/math/struct/end}{test-mathml}
255   \AssignSocketPlug{tagsupport/math/substruct/begin}{single}
256   \AssignSocketPlug{tagsupport/math/substruct/end}{single}
257   \int_set:Nn \l__luamml_pretty_int { 5 }
258   \RegisterFamilyMapping\symsymbols{oms}
259   \RegisterFamilyMapping\symletters{oml}
260   \AssignSocketPlug{tagsupport/math/mathml/write/prepare}{On}
261   \iow_new:N   \g__math_luamml_iow
262   \iow_open:Nn \g__math_luamml_iow {\c_sys_jobname_str-luamml-mathml.html}
263   \iow_now:Ne \g__math_luamml_iow { \c__math_mathml_write_init_tl }
264   \cs_new:Npn \__math_luamml_output_hook:n ##1
265     {
266       \tl_if_empty:NF \l__math_mathml_write_before_tl
267       {
268         \iow_now:Ne \g__math_luamml_iow
269         {
270           \l__math_mathml_write_before_tl
271           ##1
272           \c__math_mathml_write_after_tl
273         }
274       }
275     }
276   \__luamml_register_output_hook:N \__math_luamml_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:

```

277   \luamml_flag_process:

```

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

```

278   \AddToHook{enddocument/afterlastpage}
279   {
280     \iow_now:Ne \g__math_luamml_iow
281     { \c__math_mathml_write_final_tl }
282     \iow_close:N \g__math_luamml_iow
283   }
284 }

```

And now a key to deactivate luamml

```

285
286 \keys_define:nn { __tag / setup }
287 {
288   math/mathml/luamml .choice: ,
289   math/mathml/luamml/true .code:n = {\tl_gset:Nn \g__tag_math_luamml_tl{1}},
290   math/mathml/luamml/false .code:n = {\tl_gset:Nn \g__tag_math_luamml_tl{-1}},
291   math/mathml/luamml .default:n = true,
292   math/mathml/luamml .usage:n=preamble
293 }

```


`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

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

On (*plug*)

```
295 \socket_new_plug:nnn{tagsupport/math/mathml/write}{On}
296 {
297   \iow_now:Ne \g__math_writedummy_iow
298   {
299     \l__math_mathml_write_before_tl
300     <math></math>
301     \c__math_mathml_write_after_tl
302   }
303 }
```

And now a key to activate the socket.

```
304
305 \keys_define:nn { __tag / setup }
306 {
307   math/mathml/write-dummy .code:n =
308   {
309     \bool_gset_true:N \g__tag_math_mathml_AF_bool
310     \tl_if_exist:NF\g__math_writedummy_iow
311     {
312       \iow_new:N \g__math_writedummy_iow
313       \iow_open:Nn \g__math_writedummy_iow
314       {
315         \c_sys_jobname_str-mathml-dummy.html
316       }
317       \iow_now:Ne \g__math_writedummy_iow
318       {
319         \c__math_mathml_write_init_tl
320       }
321       \AssignSocketPlug {tagsupport/math/mathml/write/prepare}{On}
322       \AssignSocketPlug {tagsupport/math/mathml/write}{On}
323       \AddToHook{enddocument/afterlastpage}
324       {
325         \iow_now:Ne \g__math_writedummy_iow
326         { \c__math_mathml_write_final_tl }
327         \iow_close:N \g__math_writedummy_iow
328       }
329     }
330   },
331   math/mathml/write-dummy .usage:n=preamble
332 }
```

`__math_AF_process_mathml_files:`

```
333 \box_new:N\l__math_tmpa_box
334 \cs_new_protected:Npn \__math_AF_process_mathml_files:
335 {
336   \hbox_set:Nn \l__math_tmpa_box
337   {
338     \pdfdict_put:nnn { l_pdffile/Filespec }{AFRelationship} { /Supplement }
```

```

339 \pdfdict_put:nne
340 { l_pdffile }{Subtype}
341 { \pdf_name_from_unicode_e:n{application/mathml+xml} }
342 \char_set_catcode_other:N \#
343 \cs_set_eq:NN\mml \_math_AF_html_reader:w
344 \clist_map_inline:Nn \l__tag_math_mathml_files_clist
345 {
346   \file_if_exist:nTF {##1.html}
347   {
348     \typeout{Info:~reading-mathml~file~##1}
349     \file_input:n {##1.html}
350     \bool_gset_true:N\g__tag_math_mathml_AF_bool
351   }
352   {
353     \typeout{Info:~mathml~file~##1~does~not~exist}%info message
354   }
355 }
356 }
357 \bool_if:NT\g__tag_math_mathml_AF_bool
358 {
359   \typeout{Info:~Activating-mathml~support}
360   \AssignSocketPlug{tagsupport/math/struct/begin}{test-mathml}
361   \AssignSocketPlug{tagsupport/math/struct/end}{test-mathml}

```

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

```

362   \AssignSocketPlug{tagsupport/math/substruct/begin}{single}
363   \AssignSocketPlug{tagsupport/math/substruct/end}{single}
364   \AddToHook{enddocument/info}
365   {
366     \iow_term:n{MathML~statistic}
367     \iow_term:n{=====}
368     \iow_term:e{==>~\int_use:N\g__math_mathml_total_int\c_space_tl
369     MathML~fragments~read}
370     \iow_term:e{==>~\int_use:N\g__math_mathml_int\c_space_tl
371     different~MathML~fragments}
372     \iow_term:e{==>~\int_use:N\g__math_math_total_int\c_space_tl
373     math~fragments~found}
374     \iow_term:e{==>~\int_use:N\g__math_mathml_AF_found_int\c_space_tl
375     fitting~MathML~AF~found}
376     \iow_term:e{==>~\int_use:N\g__math_mathml_AF_attached_int\c_space_tl
377     MathML~AF~attached}
378   }
379 }
380 }
381 \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
382 \int_new:N \l__math_mathstyle_int
383 \int_new:N \g__math_mathchoice_int
384 \property_new:nnnn{mathstyle}{now}{-1}{\int_use:N \l__math_mathstyle_int }

```

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

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

```

\__math_tag_if_mathstyle:nn

```

```

385 \cs_new_protected:Npn \__math_tag_if_mathstyle:nn #1 #2
386   %#1 refers to label
387   %#2 is a number for the mathstyle (typically 0,2,4,6)
388   {
389     \int_set:Nn \l__math_mathstyle_int {#2}
390     \property_record:nn {#1} { mathstyle }
391     \int_compare:nNnTF { \property_ref:nn {#1}{ mathstyle } } = { #2 }
392       { \tag_resume:n{\mathchoice} }{ \tag_suspend:n{\mathchoice} }
393   }
394 \cs_generate_variant:Nn \__math_tag_if_mathstyle:nn {en}

```

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

5.7 Tagging options

```

395 \keys_define:nn { __tag / setup }
396   {
397     math/mathml/sources .clist_set:N = \l__tag_math_mathml_files_clist,
398     math/alt/use .bool_set:N = \l__tag_math_alt_bool,
399     viewer/pane/mathml .bool_set:N = \l__tag_math_mathml_pane_bool,
400     viewer/pane/mathml .initial:n = true,
401     viewer/pane/mathsource .bool_set:N = \l__tag_math_texsource_pane_bool,
402     math/mathml/AF .bool_set:N = \l__tag_math_mathml_AF_bool,
403     math/mathml/AF .initial:n = true,
404     math/tex/AF .bool_set:N = \l__tag_math_texsource_AF_bool,
405     math/tex/AF .initial:n = true
406   }

```

5.8 Sockets

5.8.1 Main inline math sockets

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

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

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

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

```

407 \socket_new:nn {tagsupport/math/inline/begin}{0}
408 \socket_new:nn {tagsupport/math/inline/end}{0}
409 \socket_new:nn {tagsupport/math/inline/formula/begin}{1} %
410 \socket_new:nn {tagsupport/math/inline/formula/end}{0}

```

MC (*plug*)

```
411 \socket_new_plug:nnn
412   {tagsupport/math/inline/begin}
413   {MC}
414   {\tag_mc_end_push:}
415 \socket_new_plug:nnn
416   {tagsupport/math/inline/end}
417   {MC}
418   {\tag_mc_begin_pop:n{}}
```

We probably will want to test different tagging recipes.

default (*plug*)

```
419 \socket_new_plug:nnn
420   {tagsupport/math/inline/formula/begin}
421   {default}
422   { \tagpdfparaOff
423     \tag_socket_use:n{math/content}
424     \tag_socket_use:n{math/struct/begin}
```

TODO: does inline math need subformula handling?

```
425     % inner formula if multiple parts (not really implemented yet)
426     \tag_socket_use:n{math/substruct/begin}
427     #1
428     \tag_socket_use:n{math/end}
429   }
430 \socket_new_plug:nnn
431   {tagsupport/math/inline/formula/end}
432   {default}
433   {
434     \socket_use:n{tagsupport/math/substruct/end}
435     \socket_use:n{tagsupport/math/struct/end}
436   }
```

5.8.2 Main display math sockets

`\tagsupport/math/display/begin (socket)` The first two sockets are meant to embed display math into the surrounding (so to close/reopen e.g. MC-chunks and P-structure). The other two implement the actual formula structure. The formula sockets are despite their naming not symmetric: the begin socket is issued after the math has started, while the end socket is after the math! The socket `\tagsupport/math/display/formula/begin` should be similar to the inline version not be used as tagging socket so that the argument, the math, is not lost.

```
437 \socket_new:nn {tagsupport/math/display/begin}{0}
438 \socket_new:nn {tagsupport/math/display/end}{0}
439 \socket_new:nn {tagsupport/math/display/formula/begin}{1} %
440 \socket_new:nn {tagsupport/math/display/formula/end}{0}
```

default (*plug*)

```
441 \socket_new_plug:nnn
442   {tagsupport/math/display/begin}
443   {default}
444   { \_tag_tool_close_P: }
445 \socket_new_plug:nnn
```

```

446 {tagssupport/math/display/end}
447 {default}
448 {
449 }

```

default (*plug*)

```

450 \socket_new_plug:nnn
451 {tagssupport/math/display/formula/begin}
452 {default}
453 {
454   \tag_socket_use:n{math/content}
455   \tag_socket_use:n{math/struct/begin}
456   \tag_socket_use:n{math/substruct/begin}
457   #1
458   \tag_socket_use:n{math/end}
459 }
460 \socket_new_plug:nnn
461 {tagssupport/math/display/formula/end}
462 {default}
463 {
464   \socket_use:n{tagssupport/math/substruct/end}
465   \socket_use:n{tagssupport/math/struct/end}
466 }

```

5.8.3 Internal sockets

\l__math_content_template_tl

The default text used as alt or actual text.

```

467 \tl_new:N\l__math_content_template_tl
468 \tl_set:Nn \l__math_content_template_tl
469   {
470     LaTeX~ formula~ starts~
471     \exp_not:N\begin{\g__math_grabbed_env_tl}
472     \c_space_tl
473     \exp_not:V\g__math_grabbed_math_tl
474     \c_space_tl
475     \exp_not:N\end{\g__math_grabbed_env_tl}
476     \c_space_tl LaTeX~ formula~ ends~
477   }

```

\l__math_texsource_template_tl

The default text used as texsource

```
478 \tl_new:N\l__math_texsource_template_tl
479 \tl_const:Nn\c__math_inline_env_tl {math}
480 \tl_set:Nn \l__math_texsource_template_tl
481   {
482     \tl_if_eq:NNTF\g__math_grabbed_env_tl\c__math_inline_env_tl
483     {
484       $
485       \exp_not:V\g__math_grabbed_math_tl
486       $
487     }
488     {
489       \exp_not:N\begin{\g__math_grabbed_env_tl}
490       \exp_not:V\g__math_grabbed_math_tl
491       \exp_not:N\end{\g__math_grabbed_env_tl}
492     }
493   }
```

`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`

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

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

`actual+source` (*plug*)

```
495 \socket_new_plug:nnn
496   {tagsupport/math/content}
497   {actual+source}
498   {
499     \tl_set:Ne\l__math_content_actual_tl
500     {
501       \l__math_content_template_tl
502     }
503     \tl_set:Ne \l__math_content_AF_source_tl
504     {
505       \l__math_texsource_template_tl
506     }
507     \tl_set:Nn \l__math_content_AF_mathml_tl {}
508     \tl_set:Nn \l__math_content_alt_tl {}
509   }
```

`alt+source` (*plug*)

```
510 \socket_new_plug:nnn
511   {tagsupport/math/content}
512   {alt+source}
513   {
514     \tl_set:Ne\l__math_content_alt_tl
```

```

515     {
516       \l__math_content_template_tl
517     }
518   \tl_set:Nc \l__math_content_AF_source_tl
519     {
520       \l__math_texsource_template_tl
521     }
522   \tl_set:Nn \l__math_content_AF_mathml_tl {}
523   \tl_set:Nn \l__math_content_actual_tl {}
524 }

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

```

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

```

\g__math_grabbed_env_tl and \g__math_grabbed_math_tl
526 \socket_new:nn {tagsupport/math/struct/begin}{0}
527 \socket_new:nn {tagsupport/math/struct/end}{0}

```

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

```

528 \socket_new_plug:nnn
529   {tagsupport/math/struct/begin}
530   {default}
531   {
532     \bool_if:NTF\l__tag_math_texsource_AF_bool
533     { \tl_set_eq:NN \l__math_content_AF_source_tmpa_tl \l__math_content_AF_source_tl }
534     { \tl_clear:N \l__math_content_AF_source_tmpa_tl }
535     \tag_struct_begin:n
536     {
537       tag=Formula,
538       texsource   = \l__math_content_AF_source_tmpa_tl,
539       title-o     = \g__math_grabbed_env_tl,
540       actualtext  = \l__math_content_actual_tl,
541       alt         = \l__math_content_alt_tl
542     }
543   }
544 \socket_new_plug:nnn
545   {tagsupport/math/struct/end}
546   {default}
547   { \tag_struct_end: }
548
549 \socket_assign_plug:nn {tagsupport/math/struct/begin}{default}
550 \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`

```

551 \cs_generate_variant:Nn \str_mdfive_hash:n {o}
552 \tl_new:N\l__math_content_hash_tl

```

we need to save the grabbed math:

```

553 \tl_new:N\l__math_grabbed_math_tl

```

the socket definition

```

554 \socket_new_plug:nnn
555 {tagssupport/math/struct/begin}
556 {test-mathml}
557 {
558   \int_gincr:N\g__math_math_total_int
559   \tl_set:Ne\l__math_content_hash_tl
560   {\str_mdfive_hash:o { \l__math_content_AF_source_tl }}
561   \tl_set_eq:NN\l__math_grabbed_math_tl\g__math_grabbed_math_tl
562   \tl_if_eq:NnTF\g__math_grabbed_env_tl {math}
563   {
564     \tl_set:Nn\l__math_attribute_class_tl{inline}
565   }
566   {
567     \tl_set:Nn\l__math_attribute_class_tl{display}
568   }
569   \bool_if:NF\l__tag_math_alt_bool
570   { \tl_set:Nn \l__math_content_alt_tl{} }

```

debugging option. TODO: hide in debug key.

```

571   \tl_if_exist:cTF { g__math_mathml_ \l__math_content_hash_tl _tl }
572   {
573     \int_gincr:N\g__math_mathml_AF_found_int
574     \bool_if:NTF \l__tag_math_mathml_AF_bool
575     {
576       \int_gincr:N\g__math_mathml_AF_attached_int
577       \typeout {Inserting-mathml~with~Hash~\l__math_content_hash_tl}
578     }
579     {
580       \typeout {Ignoring-mathml~with~Hash~\l__math_content_hash_tl}
581     }
582   }
583   {
584     \typeout{WARNING:~mathml~missing~for~hash~\l__math_content_hash_tl}
585   }
586   \socket_use:n {tagssupport/math/mathml/write/prepare}
587   \socket_use:n {tagssupport/math/mathml/write} % write hash if request
588   \bool_if:NTF\l__tag_math_texsource_AF_bool
589   { \tl_set_eq:NN \l__math_content_AF_source_tmpa_tl \l__math_content_AF_source_tl }
590   { \tl_clear:N \l__math_content_AF_source_tmpa_tl }
591   \tag_struct_begin:n
592   {
593     tag=Formula,
594     attribute-class=\l__math_attribute_class_tl, %
595     AFref =
596     \bool_if:NT\l__tag_math_mathml_AF_bool
597     {
598       \cs_if_exist_use:c {g__math_mathml_ \l__math_content_hash_tl _tl}
599     },
600     texsource = \l__math_content_AF_source_tmpa_tl, % should be after mathml AF!
601     title-o = \g__math_grabbed_env_tl, %
602     alt = \l__math_content_alt_tl
603   }
604 }

```


not really needed but looks more symmetric:

```
605 \socket_new_plug:nnn
606   {tagsupport/math/struct/end}
607   {test-mathml}
608   {
609     \tag_struct_end:
610   }
```

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

```
port/math/substruct/end (socket) 611 \socket_new:nn {tagsupport/math/substruct/begin}{0}
612 \socket_new:nn {tagsupport/math/substruct/end}{0}
```

`default (plug)`

```
613 \socket_new_plug:nnn
614   {tagsupport/math/substruct/begin}
615   {default}
616   { \grabaformulapartandstart }
617 \socket_new_plug:nnn
618   {tagsupport/math/substruct/end}
619   {default}
620   {
621     \tagmcend
622     \if@subformulas
623       \tagstructend
624     \fi
625   }
626 \socket_assign_plug:nn {tagsupport/math/substruct/begin}{default}
627 \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:

```
628 \socket_new_plug:nnn
629   {tagsupport/math/substruct/begin}
630   {single}
631   {
632     \typeout{====>subpart~splitting~deactivated}
633     \typeout{====>grabbed~math=\meaning\g__math_grabbed_math_tl}
634     \tag_mc_begin:n{ }
635   }
636 \socket_new_plug:nnn
637   {tagsupport/math/substruct/end}
638   {single}
639   { \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.

```
640 \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

```
641 \cs_new_protected:Npn __tag_math_disable:
642   {
643     \socket_assign_plug:nn {tagsupport/math/inline/begin}{noop}
644     \socket_assign_plug:nn {tagsupport/math/inline/end}{noop}
```

```

645 \socket_assign_plug:nn {tagsupport/math/inline/formula/begin}{identity}
646 \socket_assign_plug:nn {tagsupport/math/inline/formula/end}{noop}
647 \socket_assign_plug:nn {tagsupport/math/display/begin}{noop}
648 \socket_assign_plug:nn {tagsupport/math/display/end}{noop}
649 \socket_assign_plug:nn {tagsupport/math/display/formula/begin}{identity}
650 \socket_assign_plug:nn {tagsupport/math/display/formula/end}{noop}
651 }

```

(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

```

652 \cs_new_protected:Npn \_tag_math_enable:
653 {
654   \socket_assign_plug:nn {tagsupport/math/inline/begin}{MC}
655   \socket_assign_plug:nn {tagsupport/math/inline/end}{MC}
656   \socket_assign_plug:nn {tagsupport/math/inline/formula/begin}{default}
657   \socket_assign_plug:nn {tagsupport/math/inline/formula/end}{default}
658   \socket_assign_plug:nn {tagsupport/math/display/begin}{default}
659   \socket_assign_plug:nn {tagsupport/math/display/end}{default}
660   \socket_assign_plug:nn {tagsupport/math/display/formula/begin}{default}
661   \socket_assign_plug:nn {tagsupport/math/display/formula/end}{default}
662 }

```

(End of definition for `_tag_math_enable:`.)

At begin document we can activate:

```

663 \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
664 \cs_new_protected:Npn \_math_process:nn #1#2
665 {
666   \legacy_if:nF {measuring@ }
667   {
668     \tl_if_in:nnTF {#2} { \m@th }
669     { \bool_set_true:N\l_math_fakemath_bool }
670     { \tl_trim_spaces_apply:nN {#2} \_math_process_auxi:nn {#1} }
671   }
672 }
673 \cs_generate_variant:Nn \_math_process:nn { V }
674 \cs_new_protected:Npn \_math_process_auxi:nn #1#2
675 {
676   \tl_gset:Nn \g__math_grabbed_env_tl {#2}
677   \tl_gset:Nn \g__math_grabbed_math_tl {#1}
678   \_math_process_auxii:nn {#2} {#1}
679 }
680 \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

```

681 \cs_new_protected:Npn \math_processor:n #1
682 { \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` 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.

```
683 \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.

```
684 \cs_new_protected:Npn \__math_grab_dollar:n #1
685 {
```

We need to do processing first as this picks up “fake” math mode: that information is needed below.

```
686 \__math_process:nm { 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.

```
687 \bool_lazy_any:nTF
688 {
689   {\legacy_if_p:n { measuring@ }}
690   { \l__math_fakemath_bool }
691   { \tl_if_blank_p:n {#1} }
692 }
693 {
694   #1 $ % $
695 }
696 {
697   \tag_socket_use:n {math/inline/begin} %end P-MC
```

We do not use a tagging socket here, so that the argument (the math) is not lost, tagging-project issue 661.

```
698 \socket_use:nm {taggsupport/math/inline/formula/begin}{#1}
699 $ % $
700 \tag_socket_use:n {math/inline/formula/end}
701 \tag_socket_use:n {math/inline/end} % restart P-MC
702 }
703 }
```

(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.

```
704 \cs_new_protected:Npn \__math_grab_dollar_delim:w #1 $ % $
705 { \__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.

```
706 \cs_new_protected:Npn \_math_grab_dollardollar:w % $$
707 #1 $$
708 {
709   \tl_if_blank:nF {#1}
710   {
711     \_math_process:nn { equation* } {#1}
712     \tag_socket_use:n {math/display/begin}
713     \socket_use:nn{tagsupport/math/display/formula/begin}{#1}
714   }
715   $$
716 }
```

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

```
717 \cs_new_protected:Npn \_math_tag_dollardollar_display_end:
718 {
719   % \typeout{== tag dollar\dollar display end}
720   % \ShowTagging{struct-stack}
721   \para_raw_end:
```

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

```
722   \tagpdfpara0n
```

The `\postdisplaypenalty` was temporarily set to 10000 inside the display and the `\belowdisplayskip` and the `\belowdisplayshortskip` was negated, so whatever was inserted it should have been a negative skip. Whatever skip was added we pick it up value up here, so that we can correct the spacing after the tagging code was inserted.

```
723   \l_math_tmpa_skip \lastskip
724   \tag_socket_use:n{math/display/formula/end}
```

Now we add a skip without introducing a page break possibility, that should bring the current vertical position back to the point where T_EX would add the penalty and the “below skip”.

```
725   \nobreak
726   \skip_vertical:n { -\l_math_tmpa_skip } % remove the negative belowdisplayskip
```

Then we finally add the real stuff:

```
727   \penalty \postdisplaypenalty
728   \skip_vertical:n { -\l_math_tmpa_skip } % insert the correct skip
729   \@doendpe % this has no \end{...} to take care of it
730 }
731
```

(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.

```
732 \cs_new_protected:Npn \_math_grab_inline:w % \langle
733 #1 \rangle
734 {
735   \tl_if_blank:nF {#1}
736   {
737     $ #1 $
738   }
739   \bool_set_false:N \l_math_collected_bool
740 }
```

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

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

```
741 \cs_new_protected:Npn \_math_grab_eqn:w % \[
742   #1 \]
743   {
744   % \typeout{collected? = \bool_if:NTF \l_math_collected_bool {true}{false}}
745   \begin { equation* } #1 \end { equation* }
746   }
```

(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.

```
747 \tl_new:N \l_math_grabbed_tl
```

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

```
748 \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:`

```
749 \cs_new_protected:Npn \_math_grab_dollar_loop:
750   {
751   \group_begin:
752   \tl_clear:N \l_math_grabbed_tl
753   \_math_grab_loop:
754   }
755 \cs_new_protected:Npn \_math_grab_loop:
756   {
757   \peek_remove_spaces:n
758   {
759   \peek_meaning:NTF \c_group_begin_token
760     { \_math_grab_loop_group:n }
761     { \_math_grab_loop_token:N }
762   }
763   }
```

(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
764 \cs_new_protected:Npn \__math_grab_loop_group:n #1
765 { \__math_grab_loop_store:n { #1 } }
766 \cs_new_protected:Npn \__math_grab_loop_store:n #1
767 {
768   \tl_put_right:Nn \l__math_grabbed_tl {#1}
769   \__math_grab_loop:
770 }

```

(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 than a loop (cf. `collcell`).

```

\__math_grab_loop_$:
\__math_grab_loop_\\:
771 \cs_new_protected:Npn \__math_grab_loop_token:N #1
\__math_grab_loop_\begin:
772 {
\__math_grab_loop_\end:
773   \cs_if_exist_use:cF
774   { \__math_grab_loop_ \token_to_str:N #1 : }
\__math_grab_loop_\ignorespaces:
775   { \__math_grab_loop_store:n {#1} }
\__math_grab_loop_\unskip:
776 }
\__math_grab_loop_\textonly@unskip:
777 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N $ : }
778 { \__math_grab_loop_end: }
779 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N \\ : }
780 {
781   \int_compare:nNnTF \l__math_grab_env_int = 0
782   { \__math_grab_loop_newline: }
783   { \__math_grab_loop_store:n { \\ } }
784 }

```

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.

```

785 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N \begin : }
786 {
787   \int_incr:N \l__math_grab_env_int
788   \__math_grab_loop_store:n { \begin }
789 }
790 \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N \end : }
791 {
792   \int_compare:nNnTF \l__math_grab_env_int = 0
793   {
794     \__math_grab_loop_newline:
795     \end
796   }
797   {
798     \int_decr:N \l__math_grab_env_int
799     \__math_grab_loop_store:n { \end }
800   }
801 }
802 \tl_map_inline:nn { \ignorespaces \unskip \textonly@unskip }
803 {
804   \cs_new_protected:cpn { \__math_grab_loop_ \token_to_str:N #1 : }
805   { \__math_grab_loop: }

```

```
806 }
(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`.

```
807 \cs_new_protected:Npn \__math_grab_loop_newline:
808 {
809   \if_false: { \fi:
810     \cs_set_protected:Npn \cr
811       {
812         \__math_grab_loop:
813         \tex_cr:D
814       }
815   \if_false: } \fi:
816   \\\
817 }
```

(End of definition for `__math_grab_loop_newline:.`)

`__math_grab_loop_end:` Clean up and pass on.

```
818 \cs_new_protected:Npn \__math_grab_loop_end:
819 {
820   \exp_args:NNV \group_end:
821   \__math_grab_dollar:n \l__math_grabbed_tl
822 }
```

(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”

```
823 \tl_new:N \l__math_env_name_tl
```

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

```
824 \keys_define:nn { __math }
825 {
826   arg-spec .code:n =
827   {
828     \ExpandArgs { c } \DeclareDocumentCommand
829     { __math_env \l__math_env_name_tl _aux: }
830     {#1}
831     { \__math_env_forward:w }
832   }
833 }
```

$\backslash\text{math_register_env:nn}$
 $\backslash\text{math_register_env:n}$
 $\backslash\text{RegisterMathEnvironment}$

Set up to capture environment content and make available.

```

834 \cs_new_protected:Npn \math_register_env:nn #1#2
835 {
836   \tl_set:Nn \l__math_env_name_tl {#1}
837   \keys_set:nn { __math } {#2}
838   \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
839   \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
840   %
841   \ExpandArgs { nne } \RenewDocumentEnvironment {#1} { b }
842   {
843     \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
844     {
845       % \typeout{===>B1}
846     }
847     {
848       % \typeout{===>B2}
849       \cs_if_exist:cTF { __math_env_ #1 _aux: }
850       {
851         \exp_not:c { __math_env_ #1 _aux: }
852         ##1 \exp_not:N \__math_env_end: {#1}
853       }
854       { \exp_not:N \__math_process:nn {#1} {##1} }
855       \exp_not:n { \@kernel@math@registered@begin }
856       \bool_set_true:N \exp_not:N \l__math_collected_bool
857     }
858     % \exp_not:N \tracingall
859     \exp_not:c { __math_env_ #1 _begin: }
860     ##1
861     \exp_not:c { __math_env_ #1 _end: }
862     % \exp_not:N \tracingnone
863   }
864   {
865   }
866 }
867
868 \cs_new_protected:Npn \math_register_halign_env:nn #1#2
869 {
870   \tl_set:Nn \l__math_env_name_tl {#1}
871   \keys_set:nn { __math } {#2}
872   \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
873   \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
874   %
875   \ExpandArgs { nnee } \RenewDocumentEnvironment {#1} { b }
876   {
877     \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
878     {
879       % \typeout{===>B1}
880     }
881     {
882       % \typeout{===>B2}
883       \cs_if_exist:cTF { __math_env_ #1 _aux: }
884       {
885         \exp_not:c { __math_env_ #1 _aux: }
886         ##1 \exp_not:N \__math_env_end: {#1}

```



```

887     }
888     { \exp_not:N \l__math_process:nn {#1} {##1} }
889     \exp_not:n { \@kernel@math@registered@begin }
890     \bool_set_true:N \exp_not:N \l__math_collected_bool
891   }
892 % \exp_not:N \tracingall
893 \exp_not:c { __math_env_ #1 _begin: }
894 ##1
895 % \exp_not:N \tracingnone
896 }
897 {
898 \exp_not:c { __math_env_ #1 _end: }
899 }
900 }

```

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

```

901 \cs_new_protected:Npn \math_register_odd_env:nn #1#2
902 {
903   \tl_set:Nn \l__math_env_name_tl {#1}
904   \keys_set:nn { __math } {#2}
905   \cs_gset_eq:cc { __math_env_ #1 _begin: } {#1}
906   \cs_gset_eq:cc { __math_env_ #1 _end: } { end #1 }
907 %
908 \ExpandArgs { nnee } \RenewDocumentEnvironment {#1} { b }
909 {
910   \exp_not:N \bool_if:NTF \exp_not:N \l__math_collected_bool
911   {
912 % \typeout{===>B1}
913   }
914   {
915 % \typeout{===>B2}
916   \cs_if_exist:cTF { __math_env_ #1 _aux: }
917   {
918     \exp_not:c { __math_env_ #1 _aux: }
919     ##1 \exp_not:N \l__math_env_end: {#1}
920   }
921   { \exp_not:N \l__math_process:nn {#1} {##1} }
922   \exp_not:n { \@kernel@math@registered@begin }
923   \bool_set_true:N \exp_not:N \l__math_collected_bool
924   }
925 % \exp_not:N \tracingall
926 \exp_not:c { __math_env_ #1 _begin: }
927 ##1
928 }
929 {
930 \exp_not:c { __math_env_ #1 _end: }
931 % needed if we don't have $$...$$
932 % \exp_not:n { \typeout{---> @kernel@math@registered@end } }
933 \exp_not:n { \@kernel@math@registered@end }
934 }
935 }
936
937
938 % FMi: compare with block change!

```

```

939 %
940 % \DeclareRobustCommand*\begin[1]{%
941 % \UseHook{env/#1/before}%
942 % \ifundefined{#1}%
943 % {\def\reserved@a{\@latex@error{Environment #1 undefined}\@eha}}%
944 % {\def\reserved@a{\def\@currenvir{#1}%
945 % \edef\@currenvline{\on@line}%
946 % \execute@begin@hook{#1}%
947 % \csname #1\endcsname}}%
948 % \ignorefalse
949 % \begingroup
950 % \endpefalse % tmp!!! is it ok to drop this here?
951 % \reserved@a}
952
953
954 \cs_new:Npn \@kernel@math@registered@begin {
955 % \ShowTagging{struct-stack}
956 %\typeout{==>A1}\ShowTagging{struct-stack,mc-current}
957 \mode_if_vertical:TF
958 {
959 % \legacy_if:nTF { @endpe }
960 % { \legacy_if_set_false:n { @endpe } }
961 % { \__block_list_beginpar_vmode: }
962 %
963 % \typeout{==>~ at:~ \g__tag_struct_tag_tl}
964 %
965 \tag_if_active:T
966 {
967 \exp_args:Noo\str_if_eq:nmF \g__tag_struct_tag_tl { \l__tag_para_main_tag_tl }
968 {
969 % \typeout{==>A2}
970 \__block_beginpar_vmode:
971 } % needs correction!
972 }
973 }
974 {
975 % \typeout{==>A3}
976 \__tag_tool_close_P:
977 }
978 \socket_use:nn{tagsupport/math/display/formula/begin}{}
979 \tagpdfparaOff
980 % \typeout{==>MC1}\ShowTagging{mc-current}
981 }
982
983 \cs_new:Npn \@kernel@math@registered@end {
984 % \typeout{==>MC2}\ShowTagging{mc-current}
985 \para_raw_end:
986 \tagpdfparaOn
987 \socket_use:n{tagsupport/math/display/formula/end}
988 % \typeout{==>MC3}\ShowTagging{mc-current}
989 \@endpetrue
990 }
991
992 \cs_new_protected:Npn \math_register_env:n #1

```

```

993 { \math_register_env:nn {#1} { } }
994
995 \NewDocumentCommand \RegisterMathEnvironment { 0{ } m }
996 { \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`

```

997 \cs_new_protected:Npn \_math_env_forward:w #1 \_math_env_end: #2
998 { \_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
\_math_equation_begin:
\equation*
\_math_equation_star_begin:
\endequation
\_math_equation_end:
\endequation*
\_math_equation_star_end:

```

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

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

```

999 \tl_gput_right:Nn \@kernel@before@begindocument
1000 {
1001   \math_register_env:n { equation }
1002   \math_register_env:n { equation* }
1003   % at the moment register_env can only do display math
1004   %   \math_register_env:n { math }
1005   \RenewDocumentEnvironment{math} {b}{\$#1$}{ }
1006   % and this one doesn't work either
1007   %   \math_register_env:n { displaymath }
1008   \RenewDocumentEnvironment{displaymath} {b}{\[#1\]}{ }
1009 }

```

(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.

```

1010 \cs_gset_protected:Npn \( % \)
1011 {
1012   \bool_if:NTF \l_math_collected_bool
1013   {
1014     \mode_if_math:TF
1015     { \badmath }
1016     { $ }
1017   }
1018   {
1019     \_math_grab_inline:w
1020   }

```

```

1021 } % \(\
1022 \cs_gset_protected:Npn \)
1023 {
1024   \mode_if_math:TF
1025     { $ }
1026     { \@badmath }
1027 }

```

(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.

```

1028 \tl_gput_right:Nn \@kernel@before@begindocument
1029 {
1030   \cs_gset_protected:Npn \[ % \]
1031   {
1032     \__math_grab_eqn:w
1033     % \bool_if:NTF \l__math_collected_bool
1034     % { \begin { equation* } }
1035     % { \__math_grab_eqn:w }
1036   } % \[
1037   \cs_gset_protected:Npn \]
1038   {
1039     \@badmath
1040     % \bool_if:NTF \l__math_collected_bool
1041     % { \end{ equation* } }
1042     % { \@badmath }
1043   }
1044 }

```

(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.

```

1045 %\cs_gset_protected:Npn \ensuremath #1
1046 % {
1047 %   \mode_if_math:TF
1048 %     {#1}
1049 %     {
1050 %       \bool_if:NTF \l__math_collected_bool
1051 %         { \@ensuredmath {#1} }
1052 %         {
1053 %           \bool_set_true:N \l__math_collected_bool
1054 %           \__math_process:nn { math } {#1}
1055 %           \@ensuredmath {#1}
1056 %           \bool_set_false:N \l__math_collected_bool
1057 %         }
1058 %     }
1059 % }

```

(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” T_EX display. Most display math mode is actually handled elsewhere, as we have macro control.

```
1060
1061 \exp_args:No \tex_everymath:D
1062 {
1063   \tex_the:D \tex_everymath:D
1064   \bool_if:NF \l__math_collected_bool
1065   {
1066     \bool_set_true:N \l__math_collected_bool
1067     \__math_grab_dollar:w
1068   }
1069 }
1070
1071 \exp_args:No \tex_everydisplay:D
1072 {
1073   \tex_the:D \tex_everydisplay:D
1074   \iftrue % this may have to be a settable flag!
1075   % \typeout{==>~ in~ everydisplay}
```

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

```
1076     \skip_set:Nn \belowdisplayskip {-\belowdisplayskip}
1077     \skip_set:Nn \belowdisplayshortskip {-\belowdisplayshortskip}
1078     \int_set:Nn \postdisplaypenalty {10000}
1079     \group_insert_after:N \__math_tag_dollardollar_display_end:
1080   \fi
1081   \bool_if:NF \l__math_collected_bool
1082   {
1083     \bool_set_true:N \l__math_collected_bool
1084     \__math_grab_dollardollar:w
1085   }
1086 }
```

5.15 Modifying kernel environments

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

```
1087 \math_register_env:n { eqnarray }
1088 \math_register_env:n { eqnarray* }
1089 \RequirePackage{array}
1090 \tl_if_in:NnT\@tabular{$}
1091 {
1092   \def\@tabular{%
1093     \leavevmode
1094     \UseTaggingSocket{tbl/hmode/begin}%
1095     \hbox \bgroup
1096     \bool_set_true:N \l__math_collected_bool
```

```

1097 $
1098 \bool_set_false:N \l__math_collected_bool
1099 \col@sep\tabcolsep \let\dollarbegin\begin\group
1100 \let\dollarend\endgroup

```

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

```

1101 \cs_set_protected:Npn \__math_grab_dollar:w { \__math_grab_dollar_loop: }
1102 \@tabarray}
1103 }

```

`__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!

```

1104 \cs_new_eq:NN \__math_m@th: \m@th
1105 \cs_gset_protected:Npn \m@th
1106 {
1107   \bool_set_true:N \l__math_collected_bool
1108   \__math_m@th:
1109 }

```

(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.

```

1110 \tl_gput_right:Nn \@kernel@before@begindocument
1111 {
1112   \bool_set_true:N \l__math_collected_bool
1113 }
1114 \tl_gput_right:Nn \@kernel@after@begindocument
1115 {
1116   \bool_set_false:N \l__math_collected_bool
1117 }

```

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`.

```

1118
1119
1120
1121 \tl_gput_right:Nn \@kernel@before@begindocument {
1122 %
1123 \renewenvironment{gather*}{%
1124 \start@gather\st@rredtrue
1125 }
1126 {%
1127 % this redirection doesn't work if we alter "gather"!
1128 % \endgather
1129 % so replace it with its real meaning

```

```

1130 \math@cr \black@\totwidth@ \egroup
1131 $$\ignorespacesafterend
1132 }
1133 \def\common@align@ending {
1134 \math@cr \black@\totwidth@
1135 \egroup
1136 \ifingather@
1137 \restorealignstate@
1138 \egroup
1139 \nonumber
1140 \ifnum0='{\fi\iffalse}\fi
1141 \else
1142 $$%
1143 \fi
1144 \ignorespacesafterend
1145 }
1146 \renewenvironment{alignat}{%
1147 \start@align\z@\st@rredfalse
1148 }{%
1149 \common@align@ending
1150 }
1151 \renewenvironment{alignat*}{%
1152 \start@align\z@\st@rredtrue
1153 }{%
1154 \common@align@ending
1155 }
1156 \renewenvironment{xalignat}{%
1157 \start@align\@ne\st@rredfalse
1158 }{%
1159 \common@align@ending
1160 }
1161 \renewenvironment{xalignat*}{%
1162 \start@align\@ne\st@rredtrue
1163 }{%
1164 \common@align@ending
1165 }
1166 \renewenvironment{xxalignat}{%
1167 \start@align\tw@\st@rredtrue
1168 }{%
1169 \common@align@ending
1170 }
1171 \renewenvironment{align}{%
1172 \start@align\@ne\st@rredfalse\m@ne
1173 }{%
1174 \common@align@ending
1175 }
1176 \renewenvironment{align*}{%
1177 \start@align\@ne\st@rredtrue\m@ne
1178 }{%
1179 \common@align@ending
1180 }
1181 \renewenvironment{flalign}{%
1182 \start@align\tw@\st@rredfalse\m@ne
1183 }{%

```

```

1184 \common@align@ending
1185 }
1186 \renewenvironment{flalign*}{%
1187 \start@align\tw@\st@rredtrue\m@ne
1188 }{%
1189 \common@align@ending
1190 }
1191 %
1192 \renewenvironment{multline*}{\start@multline\st@rredtrue}
1193 {%
1194 \iftagsleft@ \@xp\lendmultline@ \else \@xp\rendmultline@ \fi
1195 \ignorespacesafterend
1196 }

```

Also for false?

```

1197 \def\measuring@true{\let\ifmeasuring@\iftrue\tag_suspend:n{\measuring}}
1198 %
1199 \math_register_halign_env:nn {align}{}
1200 \math_register_halign_env:nn {align*}{}
1201 \math_register_halign_env:nn {alignat}{}
1202 \math_register_halign_env:nn {alignat*}{}
1203 \math_register_halign_env:nn {flalign}{}
1204 \math_register_halign_env:nn {flalign*}{}
1205 \math_register_halign_env:nn {gather}{}
1206 \math_register_halign_env:nn {gather*}{}
1207 \math_register_halign_env:nn {multline}{}
1208 \math_register_halign_env:nn {multline*}{}
1209 \math_register_halign_env:nn {xalignat}{}
1210 \math_register_halign_env:nn {xalignat*}{}
1211 \math_register_halign_env:nn {xxalignat}{}
1212 %
1213 \@namedef{maketag @ @ @} #1{%
1214 % \typeout{--->maketag @ @ @}
1215 \ifmeasuring@
1216 \hbox{\m@th\normalfont#1}%
1217 \else
1218 \tagmccend \tagstructbegin{tag=Lbl}%
1219 \tagmccbegin{tag=Lbl}%
1220 \hbox{\m@th\normalfont#1}%
1221 \tagmccend \tagstructend \tagmccbegin}%
1222 \fi
1223 }
1224 \@namedef{math@cr @ @ @ gather}{%
1225 \ifst@rred\nonumber\fi
1226 &\relax
1227 \make@display@tag
1228 %
1229 \maybestartnewformulatag
1230 %
1231 \ifst@rred\else\global\@eqnswtrue\fi
1232 \global\advance\row@\@ne
1233 \cr
1234 }

```



```

1235 \@namedef{math@cr @ @ @ align}{%
1236   \ifst@rred\nonumber\fi
1237   \if@eqnsw \global\tag@true \fi
1238   \global\advance\row@\@ne
1239   \add@amps\maxfields@
1240   \omit
1241   \kern-\alignsep@
1242   \iftag@
1243     \setboxz@h{\@lign\strut@{\make@display@tag}}%
1244     \place@tag
1245   \fi
1246   %
1247     \maybestartnewformulatag
1248   %
1249   \ifst@rred\else\global\@eqnswtrue\fi
1250   \global\lineht@\z@
1251   \cr
1252 }

1253 \def\restore@math@cr{\@namedef{math@cr @ @ @}{
1254 %
1255   \maybestartnewformulatag
1256 %
1257   \cr}}
1258 \restore@math@cr
1259 }

```

(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.

```

1260 \cs_new:Npn \_math_split_at_nl:NN #1#2 {
1261   \tl_set:Nf \l__math_tmpa_tl {
1262     \exp_after:wN \_math_split_at_nl_first:w #1 \ \q_nil \ \s_stop }
1263   \exp_after:wN \_math_split_at_nl_aux:nnNN \l__math_tmpa_tl #1 #2
1264 }

```

and the auxiliary commands

```

1265 \cs_new:Npn \_math_split_at_nl_first:w #1 \ \ #2 \ \ #3 \s_stop
1266 {
1267   \quark_if_nil:nTF {#2}
1268     { {#1} { } }
1269     {
1270       \_math_split_chk_if_begin:ww #1 \begin \q_nil \s_mark
1271         #2 \ \ #3 \s_stop
1272     }
1273 }
1274
1275 \cs_new_protected:Npn \_math_split_at_nl_aux:nnNN #1 #2 #3 #4
1276 {
1277   \tl_gset:Nn #4 {#1}
1278   \tl_gset:Nn #3 {#2}
1279 }
1280
1281 \cs_new:Npn \_math_split_chk_if_begin:ww

```

```

1282 #1 \begin #2 #3 \s_mark #4 \ \ \q_nil \ \ \s_stop
1283 {
1284   \quark_if_nil:nTF {#2}
1285     { {#1} {#4} }
1286     {
1287       \exp_after:wN \__math_split_collect_one_end:w
1288         \__math_split_cleanup_begin_q_nil:w #1 \begin{#2} #3 \ \ #4 \s_stop
1289         { } { 1 }
1290     }
1291 }
1292
1293 \cs_new:Npn \__math_split_cleanup_begin_q_nil:w #1 \begin \q_nil {#1}
1294
1295 \cs_new:Npn \__math_split_collect_one_end:w #1 \end #2 #3 \s_stop #4 #5
1296   {
1297     \exp_args:Nf \__math_split_check_count_begins:n
1298       { \__math_split_count_begins:n { #4 #1 } } {#5}
1299       { #4 #1 \end{#2} } {#3}
1300   }
1301 \cs_new:Npn \__math_split_count_begins:n #1
1302   { \int_eval:n { 0 \__math_split_count_begins:w #1 \begin \q_nil } }
1303
1304 \cs_new:Npn \__math_split_count_begins:w #1 \begin #2
1305   { \quark_if_nil:nF {#2} { +1 \__math_split_count_begins:w } }
1306
1307 \cs_new:Npn \__math_split_check_count_begins:n #1 #2 #3 #4
1308   {
1309     \int_compare:nNnTF {#1} = {#2}
1310     {
1311       \exp_last_unbraced:Nf \__math_split_final_cleanup:nn
1312         { \__math_split:n { \__math_split_guard:n {#3} #4 } }
1313     }
1314     {
1315       \exp_args:No \use_ii_i:nn
1316         { \exp_after:wN { \int_value:w \int_eval:n { #2 + 1 } } }
1317         { \__math_split_collect_one_end:w #4 \s_stop {#3} }
1318     }
1319   }
1320 \cs_new:Npn \__math_split_final_cleanup:nn #1 #2
1321   {
1322     \exp:w \__math_split_final_cleanup:w #1
1323     \__math_split_guard:n \q_nil \s_mark { }
1324     {#2}
1325   }
1326 \cs_new:Npn \__math_split_final_cleanup:w #1 \__math_split_guard:n #2 #3 \s_mark #4
1327   {
1328     \quark_if_nil:nTF {#2}
1329     { \exp_end: { #4 #1 } }
1330     { \__math_split_final_cleanup:w #3 \s_mark { #4 #1 #2 } }
1331   }
1332
1333 \cs_new:Npn \__math_split:n #1 {
1334   \__math_split_at_nl_first:w #1 \ \ \q_nil \ \ \s_stop }
1335

```

```

1336 % this looks unused.
1337 %\NewDocumentCommand \splitnl { mm +m }
1338 % {
1339 % \tl_set:Nf \l__math_tmpa_tl { \split:n {#3} }
1340 % \show \l__math_tmpa_tl
1341 % \exp_after:wN \__splitnl_aux:nnNN \l__math_tmpa_tl #1 #2
1342 % }

```

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

`\maybestartnewformulatag`

```

1343
1344 \newif\if@subformulas
1345 \tl_new:N \result
1346
1347 \cs_new_protected:Npn\grabaformulapartandstart {
1348 \__math_split_at_nl:NN \g__math_grabbed_math_tl \result
1349 \typeout{====>first-result=\meaning\result}
1350 \typeout{====>first-tmpmathcontent=\meaning\g__math_grabbed_math_tl}
1351 \tl_if_empty:NTF \g__math_grabbed_math_tl
1352 {
1353 \typeout{====>formula~ has~ no~ subparts}
1354 \global\@subformulasfalse
1355 }
1356 {
1357 \typeout{====>formula~ has~ subparts}
1358 \global\@subformulastrue
1359 \edef\resulttitle{\g__math_grabbed_env_tl\space (part)}
1360 \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

```

1361 % alt=\result,
1362 % alt = subformula,
1363 % title-o=\resulttitle
1364 % }
1365 % }
1366 \tagmcbegin{}
1367 }
1368
1369 \cs_new_protected:Npn\grabaformulapartandmayberestart {
1370 \__math_split_at_nl:NN \g__math_grabbed_math_tl \result
1371 \typeout{====>result=\meaning\result}
1372 \typeout{====>tmpmathcontent=\meaning\g__math_grabbed_math_tl}
1373 % \tl_if_empty:NTF \g__math_grabbed_math_tl
1374 % {
1375 % \typeout{====>tmpmathcontent=empty}
1376 % }
1377 % {
1378 % \typeout{====>tmpmathcontent=not-empty}
1379 % \edef\resulttitle{\g__math_grabbed_env_tl\space (part)}
1380 % \tagstructbegin{tag=Formula,
1381 % alt=\result,
1382 % title-o=\resulttitle
1383 % }

```

```

1384 % }
1385 \tagmcbegin{}
1386 }

```

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

```

1387 \def\maybestartnewformulatag {
1388 \if@subformulas
1389 \ifmeasuring@\else
1390 %
1391 \tl_if_empty:NF \g__math_grabbed_math_tl
1392 {
1393 \tagmcbegin
1394 \tagstructend
1395 \grabaformulapartandmaybe restart
1396 }
1397 \fi
1398 \fi
1399 }

```

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

```

1400 %\AddToHook{package/breqn/after}{
1401 % \typeout{===>~ in~ hook}
1402 % \math_register_halign_env:nn {dmath}{}
1403 % \math_register_halign_env:nn {dgroup*}{}
1404 %}
1405 \ExplSyntaxOff
1406 <@@=)
1407 </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, 342
<code>\%</code>	73
<code>\(</code>	732
<code>\(</code>	<u>1010</u>
<code>\)</code>	733
<code>\)</code>	<u>1010</u>
<code>@@</code> commands:	
<code>\l_@@_collected_bool</code>	3, 7
<code>\l_@@_fakemath_bool</code>	8
<code>\[</code>	741, 1008
<code>\[</code>	9, 29, 35, 36, <u>1028</u>
<code>\</code>	242, 243, 244, 245, 246, 247, 779, 783, 816, 1262, 1265, 1271, 1282, 1288, 1334
<code>\{</code>	70
<code>\}</code>	71
<code>\]</code>	742, 1008
<code>\]</code>	9, 29, 35, 36, <u>1028</u>
A	
<code>actual+source (plug)</code>	<u>495</u>
<code>\AddToHook</code>	7, 9, 121, 208, 278, 323, 364, 381, 1400
<code>\advance</code>	1232, 1238
<code>\aftergroup</code>	28
<code>alt+source (plug)</code>	<u>510</u>
<code>\AssignSocketPlug</code>	253, 254, 255, 256, 260, 321, 322, 360, 361, 362, 363
<code>\AtBeginDocument</code>	663
B	
<code>\begin</code>	30,

	471, 489, 745, 785, 788, 940, 1034, 1270, 1282, 1288, 1293, 1302, 1304	
<code>\beginpgroup</code> . . .	69, 145, 160, 176, 949, 1099	
<code>\belowdisplay</code>	37	
<code>\belowdisplayshortskip</code>	28, 1077	
<code>\belowdisplayskip</code>	28, 37, 1076	
<code>\bgroup</code>	1095	
block internal commands:		
<code>_block_beginpar_vmode</code> :	970	
<code>_block_list_beginpar_vmode</code> : . .	961	
bool commands:		
<code>\bool_gset_true:N</code>	252, 309, 350	
<code>\bool_if:NTF</code>	62, 357, 532, 569, 574, 588, 596, 744, 843, 877, 910, 1012, 1033, 1040, 1050, 1064, 1081	
<code>\bool_lazy_any:nTF</code>	687	
<code>\bool_new:N</code> 10, 11, 37, 38, 39, 40, 41, 42		
<code>\bool_set_false:N</code> 739, 1056, 1098, 1116		
<code>\bool_set_true:N</code>	669, 856, 890, 923, 1053, 1066, 1083, 1096, 1107, 1112	
bool internal commands:		
<code>\l_math_collected_bool</code>	9, 10, 739, 744, 843, 856, 877, 890, 910, 923, 1012, 1033, 1040, 1050, 1053, 1056, 1064, 1066, 1081, 1083, 1096, 1098, 1107, 1112, 1116	
<code>\l_math_fakemath_bool</code> 9, 11, 669, 690		
box commands:		
<code>\box_new:N</code>	333	
box internal commands:		
<code>\l_math_tmpa_box</code>	333, 336	
<code>\boxed</code>	7	
C		
char commands:		
<code>\char_set_catcode_other:N</code>	70, 71, 72, 73, 342	
clist commands:		
<code>\clist_map_inline:Nn</code>	344	
<code>\clist_new:N</code>	50	
<code>\clist_put_right:Nn</code>	51	
<code>\cr</code>	810, 1233, 1251, 1257	
cs commands:		
<code>\cs_generate_variant:Nn</code> 394, 551, 673		
<code>\cs_gset_eq:NN</code>	838, 839, 872, 873, 905, 906	
<code>\cs_gset_protected:Npn</code>	1010, 1022, 1030, 1037, 1045, 1105	
<code>\cs_if_eq:NNTF</code>	145, 160, 176	
<code>\cs_if_exist:NTF</code>	849, 883, 916	
<code>\cs_if_exist_use:N</code>	598	
<code>\cs_if_exist_use:NTF</code>	773	
<code>\cs_new:Npn</code>	264, 954, 983, 1260, 1265, 1281, 1293, 1295, 1301, 1304, 1307, 1320, 1326, 1333	
<code>\cs_new_eq:NN</code>	1104	
<code>\cs_new_protected:Npn</code>	22, 53, 68, 76, 250, 334, 385, 641, 652, 664, 674, 680, 681, 683, 684, 704, 706, 717, 732, 741, 749, 755, 764, 766, 771, 777, 779, 785, 790, 804, 807, 818, 834, 868, 901, 992, 997, 1275, 1347, 1369	
<code>\cs_set:Npn</code>	123, 135	
<code>\cs_set_eq:NN</code>	343	
<code>\cs_set_protected:Npn</code> 682, 810, 1101		
<code>\csname</code>	947	
D		
<code>\DeclareDocumentCommand</code>	828	
<code>\DeclareMathEnvironment</code>	8	
<code>\DeclareRobustCommand</code>	940	
<code>\def</code> . 943, 944, 1092, 1133, 1197, 1253, 1387		
<code>default (plug)</code>	419, 441, 450, 528, 613	
<code>\displaylines</code>	8	
E		
<code>\edef</code>	945, 1359, 1379	
<code>\egroup</code>	127, 130, 1130, 1135, 1138	
<code>\else</code>	1141, 1194, 1217, 1231, 1249, 1389	
<code>\end</code>	30, 475, 491, 729, 745, 790, 795, 799, 1041, 1295, 1299	
end internal commands:		
<code>_math_tag_dollardollar_</code>		
<code>display_end</code>	37	
<code>\endarray</code>	123	
<code>\endcsname</code>	947	
<code>\endequation</code>	999	
<code>\endequation*</code>	999	
<code>\endgather</code>	1128	
<code>\endgroup</code>	77, 1100	
<code>\ensuremath</code>	8, 1045	
<code>\equalign</code>	8	
<code>\equation</code>	999	
<code>\equation*</code>	999	
<code>\everydisplay</code>	37	
<code>\everymath</code>	3, 36, 37	
exp commands:		
<code>\exp:w</code>	1322	
<code>\exp_after:wN</code>	1262, 1263, 1287, 1316, 1341	
<code>\exp_args:Nf</code>	1297	
<code>\exp_args:NNV</code>	820	
<code>\exp_args:No</code>	1061, 1071, 1315	
<code>\exp_args:Noo</code>	967	
<code>\exp_end:</code>	1329	

legacy commands:			
<code>\legacy_if:nTF</code>	666, 959		
<code>\legacy_if_p:n</code>	689		
<code>\legacy_if_set_false:n</code>	960		
<code>\let</code>	1099, 1100, 1197		
<code>\tlabmathdate</code>	4		
<code>\tlabmathversion</code>	5		
luamml commands:			
<code>\luamml_flag_process:</code>	277		
<code>\luamml_flag_save:nn</code>	151, 166, 182		
luamml internal commands:			
<code>__luamml_array_finalize_array:</code>	131		
<code>__luamml_array_finalize_col:w</code>	153, 168, 184		
<code>__luamml_array_init_col:</code>	149, 164, 180		
<code>__luamml_array_save_array:</code>	126		
<code>\l__luamml_pretty_int</code>	257		
<code>__luamml_register_output_hook:N</code>	276		
M			
math commands:			
<code>\math_processor:n</code>	3, 681, 681		
<code>\math_register_env:n</code>	3, 834, 992, 1001, 1002, 1004, 1007, 1087, 1088		
<code>\math_register_env:nn</code>	3, 834, 834, 993, 996		
<code>\math_register_halign_env:nn</code>	868, 1199, 1200, 1201, 1202, 1203, 1204, 1205, 1206, 1207, 1208, 1209, 1210, 1211, 1402, 1403		
<code>\math_register_odd_env:nn</code>	901		
math internal commands:			
<code>__math_AF_html_reader:w</code>	68, 343		
<code>__math_AF_html_reader_verb:w</code>	74, 76		
<code>__math_AF_mml:n</code>	53, 53, 78		
<code>__math_AF_process_mathml_files:</code>	333, 334, 381		
<code>__math_amsmath_align@:nn</code>	1118		
<code>__math_amsmath_gather@:n</code>	1118		
<code>__math_amsmath_multline@:n</code>	1118		
<code>__math_env_end:</code>	852, 886, 919, 997		
<code>__math_env_forward:w</code>	831, 997, 997		
<code>__math_equation_begin:</code>	999		
<code>__math_equation_end:</code>	999		
<code>__math_equation_star_begin:</code>	999		
<code>__math_equation_star_end:</code>	999		
<code>__math_grab_dollar:n</code>	683, 684, 705, 821		
<code>__math_grab_dollar:w</code>	683, 683, 1067, 1101		
<code>__math_grab_dollar_delim:w</code>	683, 704, 704		
<code>__math_grab_dollar_loop:</code>	749, 749, 1101		
<code>__math_grab_dollardollar:w</code>	706, 706, 1084		
<code>__math_grab_eqn:w</code>	741, 741, 1032, 1035		
<code>__math_grab_inline:w</code>	732, 732, 1019		
<code>__math_grab_loop:</code>	749, 753, 755, 769, 805, 812		
<code>__math_grab_loop_:</code>	771		
<code>__math_grab_loop_\\:</code>	771		
<code>__math_grab_loop_\\begin:</code>	771		
<code>__math_grab_loop_\\end:</code>	771		
<code>__math_grab_loop_\\ignorespaces:</code>	771		
<code>__math_grab_loop_\\textonly@unskip:</code>	771		
<code>__math_grab_loop_\\unskip:</code>	771		
<code>__math_grab_loop_end:</code>	778, 818, 818		
<code>__math_grab_loop_group:n</code>	760, 764, 764		
<code>__math_grab_loop_newline:</code>	782, 794, 807, 807		
<code>__math_grab_loop_store:n</code>	764, 765, 766, 775, 783, 788, 799		
<code>__math_grab_loop_token:N</code>	761, 771, 771		
<code>__math_luamml_activate_write:</code>	213, 225, 250		
<code>__math_luamml_output_hook:n</code>	264, 276		
<code>__math_m@th:</code>	1104, 1104, 1108		
<code>__math_process:nn</code>	664, 664, 673, 686, 711, 854, 888, 921, 998, 1054		
<code>__math_process_auxi:nn</code>	664, 670, 674		
<code>__math_process_auxii:nn</code>	664, 678, 680, 682		
<code>__math_split:n</code>	1312, 1333		
<code>__math_split_at_nl:NN</code>	1260, 1260, 1348, 1370		
<code>__math_split_at_nl_aux:nnNN</code>	1263, 1275		
<code>__math_split_at_nl_first:w</code>	1262, 1265, 1334		
<code>__math_split_check_count_-begins:nnnn</code>	1297, 1307		
<code>__math_split_chk_if_begin:ww</code>	1270, 1281		
<code>__math_split_cleanup_begin_q_-nil:w</code>	1288, 1293		
<code>__math_split_collect_one_end:w</code>	1287, 1295, 1317		
<code>__math_split_count_begins:n</code>	1298, 1301		
<code>__math_split_count_begins:w</code>	1302, 1304, 1305		

<code>\socket_new:nn</code>	98, 294, 407, 408, 409, 410, 437, 438, 439, 440, 494, 526, 527, 611, 612, 640
<code>\socket_new_plug:nnn</code>	99, 295, 411, 415, 419, 430, 441, 445, 450, 460, 495, 510, 528, 544, 554, 605, 613, 617, 628, 636
<code>\socket_use:n</code>	434, 435, 464, 465, 586, 587, 987
<code>\socket_use:nn</code>	698, 713, 978
Sockets:	
<code>tagsupport/math/content</code>	494
<code>tagsupport/math/display/begin</code> ..	437
<code>tagsupport/math/display/end</code> ...	437
<code>tagsupport/math/display/formula/begin</code>	437
<code>tagsupport/math/display/formula/end</code>	437
<code>tagsupport/math/end</code>	640
<code>tagsupport/math/inline/begin</code> ..	407
<code>tagsupport/math/inline/end</code>	407
<code>tagsupport/math/inline/formula/begin</code>	407
<code>tagsupport/math/inline/formula/end</code>	407
<code>tagsupport/math/mathml/write</code> ..	294
<code>tagsupport/math/mathml/write/prepare</code>	98
<code>tagsupport/math/struct/begin</code> ...	526
<code>tagsupport/math/struct/end</code>	526
<code>tagsupport/math/substruct/begin</code> ..	611
<code>tagsupport/math/substruct/end</code> ..	611
<code>\space</code>	4, 5, 1359, 1379
split commands:	
<code>\split:n</code>	1339
<code>\splitnl</code>	1337
splitnl internal commands:	
<code>_splitnl_aux:nnNN</code>	1341
str commands:	
<code>\c_backslash_str</code>	108
<code>\str_case:nn</code>	210
<code>\str_if_eq:nnTF</code>	967
<code>\str_mdfive_hash:n</code>	551, 560
<code>\str_new:N</code>	16
<code>\str_replace_all:Nnn</code>	102, 103
<code>\str_set:Nn</code>	101
str internal commands:	
<code>\l_math_tmpa_str</code>	9, 16, 101, 102, 103, 110
<code>\symletters</code>	259
<code>\symsymbols</code>	258
sys commands:	
<code>\sys_if_engine luatex:TF</code>	116
<code>\c_sys_jobname_str</code>	52, 262, 315
	T
<code>\tabcolsep</code>	1099
tag commands:	
<code>\tag_if_active:TF</code>	24, 663, 965
<code>\tag_mc_begin:n</code>	634
<code>\tag_mc_begin_pop:n</code>	418
<code>\tag_mc_end:</code>	26, 639
<code>\tag_mc_end_push:</code>	26, 414
<code>\tag_resume:n</code>	392
<code>\tag_socket_use:n</code>	423, 424, 426, 428, 454, 455, 456, 458, 697, 700, 701, 712, 724
<code>\tag_struct_begin:n</code>	4, 535, 591
<code>\tag_struct_end:</code>	29, 547, 609
<code>\tag_suspend:n</code>	392, 1197
tag internal commands:	
<code>_tag_check_para_end_show:nn</code> ...	28
<code>_tag_gincr_para_end_int:</code>	27
<code>\l_tag_math_alt_bool</code> ..	11, 42, 398, 569
<code>_tag_math_disable:</code>	641, 641
<code>_tag_math_enable:</code> ...	652, 652, 663
<code>\g_tag_math_luamml_tl</code>	11, 43, 44, 210, 289, 290
<code>\g_tag_math_mathml_AF_bool</code>	11, 40, 252, 309, 350, 357
<code>\l_tag_math_mathml_AF_bool</code>	11, 39, 402, 574, 596
<code>\l_tag_math_mathml_files_clist</code> .	11, 50, 51, 344, 397
<code>\l_tag_math_mathml_pane_bool</code> ...	11, 41, 62, 399
<code>\l_tag_math_texsource_AF_bool</code> ..	11, 37, 404, 532, 588
<code>\l_tag_math_texsource_pane_bool</code>	11, 38, 401
<code>\l_tag_para_main_tag_tl</code>	967
<code>\g_tag_struct_tag_tl</code>	963, 967
<code>_tag_tool_close_P:</code> ..	22, 22, 444, 976
<code>\tagmcbegin</code>	1219, 1221, 1366, 1385
<code>\tagmchend</code>	621, 1218, 1221, 1393
<code>\tagpdfparaOff</code>	422, 979
<code>\tagpdfparaOn</code>	722, 986
<code>\tagpdfsetup</code>	33, 246, 248
<code>\tagstructbegin</code>	1218, 1360, 1380
<code>\tagstructend</code>	623, 1221, 1394
<code>tagsupport/math/content (socket)</code> ...	494
<code>tagsupport/math/display/begin (socket)</code>	437
<code>tagsupport/math/display/end (socket)</code>	437
<code>tagsupport/math/display/formula/begin</code> (socket)	437
<code>tagsupport/math/display/formula/end</code> (socket)	437
<code>tagsupport/math/end (socket)</code>	640

tagsupport/math/inline/begin (socket)	407	\@subformulastrue	1358
tagsupport/math/inline/end (socket)	407	\@tabarray	1102
tagsupport/math/inline/formula/begin (socket)	407	\@tabular	1090, 1092
tagsupport/math/inline/formula/end (socket)	407	\@tempcnta	138
tagsupport/math/mathml/write (socket)	294	\@xp	1194
tagsupport/math/mathml/write/prepare (socket)	98	\add@amps	1239
tagsupport/math/struct/begin (socket)	526	\align@	1118
tagsupport/math/struct/end (socket)	526	\alignsep@	1241
tagsupport/math/substruct/begin (socket)	611	\ar@align@mcCell	191
tagsupport/math/substruct/end (socket)	611	\ar@mcCellbox	188
tbl commands:		\black@	1130, 1134
\tbl_crcr:n	125	\col@sep	1099
\tbl_restore_outer_cell_data:	129	\common@align@ending	1133, 1149, 1154, 1159, 1164, 1169, 1174, 1179, 1184, 1189
test-mathml (plug)	551	\count@	138
TeX and L ^A T _E X 2 _ε commands:		\cr	29, 31
\@addtopreamble	140	\d@llarbegin	144, 145, 159, 160, 175, 176, 1099
\@arrayright	132	\d@llarend	147, 152, 162, 167, 178, 183, 1100
\@badmath	1015, 1026, 1039, 1042	\do@row@strut	155, 170, 186, 192, 197, 202
\@chnum	141	\gather@	1118
\@classx	137	\halign	31
\@classz	135	\if@eqnsw	1237
\@currenvir	944	\if@subformulas	622, 1344, 1388
\@currvline	945	\ifingather@	1136
\@doendpe	729	\ifmeasuring@	3, 8, 1197, 1215, 1389
\@eha	943	\ifst@rred	1225, 1231, 1236, 1249
\@endpbox	190, 196, 201	\iftag@	1242
\@endpefalse	950	\iftagsleft@	1194
\@endpetrue	989	\insert@column	146, 150, 161, 165, 177, 181
\@ensuredmath	1051, 1055	\insert@pcolumn	189, 195, 200
\@eqnswtrue	1231, 1249	\lendmultline@	1194
\@execute@begin@hook	946	\lineht@	1250
\@ifpackageloaded	223	\m@ne	1172, 1177, 1182, 1187
\@ifundefined	942	\m@th	7–9, 27, 38, 668, 1104, 1216, 1220
\@ignorefalse	948	\make@display@tag	1227, 1243
\@iiiparbox	7	\math@cr	1130, 1134
\@kernel@after@begindocument	1114	\maxfields@	1239
\@kernel@before@begindocument	999, 1028, 1110, 1121	\measuring@true	1197
\@kernel@math@registered@begin	855, 889, 922, 954	\multline@	1118
\@kernel@math@registered@end	933, 983	\on@line	945
\@latex@error	943	\place@tag	1244
\@lign	1243	\prepnext@tok	139, 205
\@namedef	1213, 1224, 1235, 1253	\rendmultline@	1194
\@ne	1157, 1162, 1172, 1177, 1232, 1238	\reserved@a	943, 944, 951
\@nextchar	188, 194, 199	\restore@math@cr	1253, 1258
\@preamble	133	\restorealignstate@	1137
\@startpbox	188, 194, 199	\row@	1232, 1238
\@subformulasfalse	1354	\setboxz@h	1243
		\split@tag	38

<code>\st@rredfalse</code>	1147, 1157, 1172, 1182	<code>\l__math_content_AF_mathml_tl</code>	21, 507, 522
<code>\st@rredtrue</code>	1124, 1152, 1162, 1167, 1177, 1187, 1192	<code>\l__math_content_AF_source_tl</code>	19, 101, 503, 518, 533, 560, 589
<code>\start@align</code>	1147, 1152, 1157, 1162, 1167, 1172, 1177, 1182, 1187	<code>\l__math_content_AF_source_tmpa_tl</code>	20, 533, 534, 538, 589, 590, 600
<code>\start@gather</code>	1124	<code>\l__math_content_AF_tl</code>	10
<code>\start@multline</code>	1192	<code>\l__math_content_alt_tl</code>	10, 17, 508, 514, 541, 570, 602
<code>\strut@</code>	1243	<code>\l__math_content_hash_tl</code>	112, 552, 559, 571, 577, 580, 584, 598
<code>\tag@true</code>	1237	<code>\l__math_content_template_tl</code>	21, 467, 468, 501, 516
<code>\textonly@unskip</code>	802	<code>\l__math_env_name_tl</code>	31, 823, 829, 836, 870, 903
<code>\totwidth@</code>	1130, 1134	<code>\g__math_grabbed_env_tl</code>	9, 22, 23, 12, 471, 475, 482, 489, 491, 539, 562, 601, 676, 1359, 1379
<code>\tw@</code>	1167, 1182, 1187	<code>\g__math_grabbed_math_tl</code>	9, 22, 23, 13, 473, 485, 490, 561, 633, 677, 1348, 1350, 1351, 1370, 1372, 1373, 1391
<code>\z@</code>	1147, 1152, 1250	<code>\l__math_grabbed_math_tl</code>	553, 561
tex commands:		<code>\l__math_grabbed_tl</code>	29, 747, 752, 768, 821
<code>\tex_cr:D</code>	813	<code>\c__math_inline_env_tl</code>	479, 482
<code>\tex_everydisplay:D</code>	1071, 1073	<code>\c__math_mathml_write_after_tl</code>	80, 272, 301
<code>\tex_everymath:D</code>	1061, 1063	<code>\l__math_mathml_write_before_tl</code>	80, 104, 266, 270, 299
<code>\tex_the:D</code>	1063, 1073	<code>\c__math_mathml_write_final_tl</code>	80, 281, 326
<code>\text</code>	18	<code>\c__math_mathml_write_init_tl</code>	80, 263, 319
tl commands:		<code>\l__math_texsource_template_tl</code>	22, 478, 480, 505, 520
<code>\c_space_tl</code>	108, 368, 370, 372, 374, 376, 472, 474, 476	<code>\l__math_tmpa_tl</code>	9, 14, 61, 63, 65, 1261, 1263, 1339, 1340, 1341
<code>\tl_clear:N</code>	534, 590, 752	token commands:	
<code>\tl_const:Nn</code>	80, 88, 94, 479	<code>\token_to_str:N</code>	774, 777, 779, 785, 790, 804
<code>\tl_gput_right:Nn</code>	999, 1028, 1110, 1114, 1121	<code>\tracingall</code>	858, 892, 925
<code>\tl_gset:Nn</code>	44, 289, 290, 676, 677, 1277, 1278	<code>\tracingnone</code>	862, 895
<code>\tl_gset_eq:NN</code>	65	<code>\typeout</code>	348, 353, 359, 577, 580, 584, 632, 633, 719, 744, 845, 848, 879, 882, 912, 915, 932, 956, 963, 969, 975, 980, 984, 988, 1075, 1214, 1349, 1350, 1353, 1357, 1371, 1372, 1375, 1378, 1401
<code>\tl_if_blank:nTF</code>	709, 735		
<code>\tl_if_blank_p:n</code>	691		
<code>\tl_if_empty:NNTF</code>	266, 1351, 1373, 1391		
<code>\tl_if_eq:NNTF</code>	482		
<code>\tl_if_eq:NnTF</code>	562		
<code>\tl_if_exist:NNTF</code>	57, 310, 571		
<code>\tl_if_in:NnTF</code>	1090		
<code>\tl_if_in:nnTF</code>	668		
<code>\tl_map_inline:nn</code>	802		
<code>\tl_new:N</code>	12, 13, 14, 17, 18, 19, 20, 21, 32, 43, 64, 87, 467, 478, 552, 553, 747, 823, 1345		
<code>\tl_put_right:Nn</code>	768		
<code>\tl_set:Nn</code>	104, 468, 480, 499, 503, 507, 508, 514, 518, 522, 523, 559, 564, 567, 570, 836, 870, 903, 1261, 1339		
<code>\tl_set_eq:NN</code>	533, 561, 589		
<code>\tl_to_str:n</code>	246, 248		
<code>\tl_trim_spaces_apply:nN</code>	670		
tl internal commands:			
<code>\l__math_attribute_class_tl</code>	32, 564, 567, 594		
<code>\l__math_content_actual_tl</code>	10, 18, 499, 523, 540		
		U	
		<code>\unskip</code>	802
		use commands:	
		<code>\use_ii_i:nn</code>	1315

<code>\UseHook</code>	941		
			V
		<code>\vbox</code>	188, 199
		<code>\vcenter</code>	7
<code>\UseTaggingSocket</code>	128, 1094	<code>\vtop</code>	194