

Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/09/17 v0.4

Abstract

Warning! This package is experimental and subject to change without regard for backwards compatibility. Performance issues may be encountered until algorithms are refined.

Contents

1	Introduction	2	5.2	Overcoming <code>\@on-</code> <code>lypreamble</code>	16
2	Specification	2	5.3	Other things	17
2.1	Using multiple fonts	3	6	Fundamentals	18
2.2	Script and <code>scriptscript</code> fonts/features	3	6.1	Enlarging the number of maths families	18
3	Maths input	3	6.2	<code>\DeclareMathSymbol</code> for unicode ranges	18
3.1	Miscellanea	4	6.3	The main <code>\setmathfont</code> macro	20
4	Package options	5	6.4	(Big) operators	28
4.1	Math ‘style’	5	6.5	Radicals	31
4.2	Bold switching	6	6.6	Delimiters	32
4.3	Symbols requiring spe- cial attention	7	6.7	Maths accents	34
I	The unicode-math pack- age	8	7	Font features	35
5	Things we need	10	7.1	OpenType maths font features	36
5.1	Package options	13	7.2	Script and <code>scriptscript</code> font options	36
			7.3	Range processing	36

8 Maths alphabets mapping definitions	44	the NFSS	64
8.1 Bold alphabets' character mappings	49	A.1 Overview	64
8.2 Definitions of the math symbols	55	III X_YTeX math font dimensions	65
9 Epilogue	55	IV Some manner of unit testing	70
II stix table data extraction	63	B The regular weight alphabets	71
A Documenting maths support in		C The bold alphabets	72

1 Introduction

This document describes the unicode-math package, which is an *experimental* implementation of a macro to unicode glyph encoding for mathematical characters. Its intended use is for X_YTeX, although it is conjectured that some effort could be spent to create a cross-format package that would also work with LuaTeX.

2 Specification

This section will turn into 'User Interface' in time, presumably.

In the ideal case, a single unicode font will contain all maths glyphs we need. Barbara Beeton's stix table provides the mapping between unicode maths glyphs and macro names (all 3298 — or however many — of them!). A single command

```
\setmathfont[{font features}]{{font name}}
```

would implement this for every every symbol and alphabetic variant. That means x to x , ξ to ξ , \leq to \leq , etc., \mathcal{H} to \mathcal{H} and so on, all for unicode glyphs within a single font.

Furthermore, this package should deal well with unicode characters for maths input, as well. This includes using literal Greek letters in formulae, resolving to upright or italic depending on preference.

Finally, maths versions must also be provided for. While I guess version selection in L^ATeX will remain the same, the specification for choosing the version fonts will probably be an optional argument:

```
\setmathfont[Version=Bold,{font features}]{{font name}}
```

This has not been implemented yet.

Instances above of

```
[{font features}]{{font name}}
```

follow from my fontspec package, and therefore any additional *font features* specific to maths fonts will hook into fontspec’s methods.

2.1 Using multiple fonts

There will probably be few cases where a single unicode maths font suffices (simply due to glyph coverage). The upcoming `stix` font comes to mind as a possible exception. It will therefore be necessary to delegate specific unicode ranges of glyphs to separate fonts. This syntax will also hook into the fontspec font feature processing:

```
\setmathfont[Range=<unicode range>,<font features>]{<font name>}
```

where *unicode range* is a comma-separated list of unicode slots and ranges such as {27D0–27EB, 27FF, 295B–297F}. Furthermore, preset names ranges could be used, such as `MiscMathSymbolsA`, with such ranges based on unicode chunks. The amount of optimisation required here to achieve acceptable performance has yet to be determined. Techniques such as saving out unicode subsets based on *unicode range* data to be `\input` in the next \LaTeX run are a possibility, but at this stage, performance without such measures seems acceptable.

2.2 Script and scriptscript fonts/features

Cambria Math uses OpenType font features to activate smaller optical sizes for `scriptsize` and `scriptscriptsize` symbols (the *B* and *C*, respectively, in A_{BC}).

Other fonts will possibly use entirely separate fonts. Both of these options must be taken into account. I hope this will be mostly automatic from the users’ points of view. The `+ssty` feature can be detected and applied automatically, and appropriate optical size information embedded in the fonts will ensure this latter case. Fine tuning should be possible automatically with fontspec options. We might have to wait until `MnMath`, for example, before we really know.

3 Maths input

\XeTeX ’s unicode support allows maths input through two methods. Like classical \TeX , macros such as `\alpha`, `\sum`, `\pm`, `\leq`, and so on, provide verbose access to the entire repertoire of characters defined by unicode. The literal characters themselves may be used instead, for more readable input files.

: TODO : describe alphabet inputs

A 0 1 2 3 4 5 6 7 8 9 + - = () i n Z

Figure 1: The unicode superscripts supported as input characters. These are the literal glyphs from Charis SIL, not the output seen when used for maths input. The ‘A’ and ‘Z’ are to provide context for the size and location of the superscript glyphs.

A 0 1 2 3 4 5 6 7 8 9 + - = () a e i o r u v x β γ ρ φ χ Z

Figure 2: The unicode subscripts supported as input characters. See note from figure 1.

3.1 Miscellanea

3.1.1 Primes

Primes (x') may be input in several ways. You may use any combination of ascii straight quote (‘), unicode prime (’), and `\prime`; when multiple primes occur next to each other, they chain together to form double, triple, or quadruple primes if the font contains pre-drawn glyphs. These may also be accessed with `\primedouble`, `\primetripel`, and `\primequadruple`.

If the font does not contain the pre-drawn glyphs or more than four primes are used, the single prime glyph is used multiple times with a negative kern to get the spacing right. There is no user interface to adjust this negative kern yet (because I haven’t decided what it should look like); if you need to, write something like this:

```
\ExplSyntaxOn
\muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }
\ExplSyntaxOff
```

3.1.2 Unicode subscripts and superscripts

You may, if you wish, use unicode subscripts and superscripts in your source document. For basic expressions, the use of these characters can make the input more readable. Adjacent sub- or super-scripts will be concatenated into a single expression.

The range of subscripts and superscripts supported by this package are shown in figures 1 and 2. Please request more if you think it is appropriate.

3.1.3 Vertical bar ‘|’

3.1.4 Colon ‘:’

3.1.5 Normalising some input characters

I believe all variant forms should be used as legal input that is normalised to a consistent output glyph, because we want to be fault-tolerant in the input. Here are the duplicates:

U+251: LATIN SMALL LETTER ALPHA U+25B: LATIN SMALL LETTER EPSILON U+263: LATIN SMALL LETTER GAMMA U+269: LATIN SMALL LETTER IOTA U+278: LATIN SMALL LETTER PHI U+28A: LATIN SMALL LETTER UPSILON U+190: LATIN CAPITAL LETTER EPSILON U+194: LATIN CAPITAL LETTER GAMMA U+196: LATIN CAPITAL LETTER IOTA U+1B1: LATIN CAPITAL LETTER UPSILON

4 Package options

4.1 Math ‘style’

Classically, $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ uses italic lowercase Greek letters and *upright* uppercase Greek letters for variables in mathematics. This is contrary to the ISO standards of using italic forms for both upper- and lowercase. Furthermore, the French (contrary again, *quelle surprise*) have been known to use upright uppercase *Latin* letters as well as upright upper- and lowercase Greek.

The unicode-math package accommodates these possibilities with an interface heavily inspired by Walter Schmidt’s `lucimatx` package: a package option `math-style` that takes one of three arguments: `TeX`, `ISO`, or `French` (case *in*-sensitive).

The philosophy behind the interface to the mathematical alphabet symbols lies in $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ’s attempt of separating content and formatting. Because input source text may come from a variety of places, the upright and ‘mathematical’ italic Latin and Greek alphabets are *unified* from the point of view of having a specified meaning in the source text. That is, to get a mathematical ‘*x*’, either the `ascii` (‘keyboard’) letter `x` may be typed, or the actual unicode character may be used. Similarly for Greek letters. The upright or italic forms are then chosen based on the `math-style` package option.

If glyphs are desired that do not map as per the package option (for example, an upright ‘*g*’ is desired but typing `g` yields ‘*g*’), *markup* is required to specify this; to follow from the example: `\mathup{g}`. Maths alphabets commands such as `\mathup` are detailed later.

Alternative interface However, some users may not like this convention. For them, an upright `x` is an upright ‘*x*’ and that’s that. (This will be the case when

Table 1: Effects of the `math-style` package option.

Package option	Example	
	Latin	Greek
<code>math-style=ISO</code>	(a, z, B, X)	$(\alpha, \beta, \Gamma, \Xi)$
<code>math-style=TeX</code>	(a, z, B, X)	$(\alpha, \beta, \Gamma, \Xi)$
<code>math-style=French</code>	(a, z, B, X)	$(\alpha, \beta, \Gamma, \Xi)$

obtaining source text from copy/pasting PDF or Microsoft Word documents, for example.) For these users, the `literal` option to `math-style` will effect this behaviour.

The `math-style` options' effects are shown in brief in table 1. Figure 3 on page 9 shows every character under the effect of this package option.

4.2 Bold switching

Similar as in the previous section, ISO standards differ somewhat to $\mathrm{T}_{\mathrm{E}}\mathrm{X}$'s conventions (and classical typesetting) for 'boldness' in mathematics. In the past, it has been customary to use bold *upright* letters to denote things like vectors and matrices. For example, $\mathbf{M} = (M_x, M_y, M_z)$. Presumably, this was due to the relatively scarcity of bold italic fonts in the pre-digital typesetting era. It has been suggested that *italic* bold symbols are used nowadays instead.

Bold Greek letters have simply been bold variant glyphs of their regular weight, as in $\boldsymbol{\xi} = (\xi_r, \xi_\varphi, \xi_\theta)$. Confusingly, the syntax in $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ has been different for these two examples: `\mathbf{M}` in the former ('**M**'), and `\bm` (or `\boldsymbol`, deprecated) in the latter ('***ξ***').

In `unicode-math`, the `\mathbf{M}` command works directly with both Greek and Latin maths alphabet characters and depending on package option either switches to upright for Latin letters (`bold-style=TeX`) as well or keeps them italic (`bold-style=ISO`).

To match the package options for non-bold characters, for `bold-style=French` all bold characters are upright, and `bold-style=literal` does not change the upright/italic shape of the letter.

Upright and italic bold mathematical letters input as direct unicode characters are normalised with the same rules. For example, with `bold-style=TeX`, a literal bold italic latin character will be typeset upright.

Note that `bold-style` is independent of `math-style`, although if the former is not specified then sensible defaults are chosen based on the latter.

The `bold-style` options' effects are shown in brief in table 2. Figure 4 on page 9 shows every character under the effect of this package option.

Table 2: Effects of the bold-style package option.

Package option	Example	
	Latin	Greek
<code>bold-style=ISO</code>	$(\mathbf{a}, \mathbf{z}, \mathbf{B}, \mathbf{X})$	$(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Gamma}, \boldsymbol{\Xi})$
<code>bold-style=TeX</code>	$(\mathbf{a}, \mathbf{z}, \mathbf{B}, \mathbf{X})$	$(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Gamma}, \boldsymbol{\Xi})$
<code>bold-style=French</code>	$(\mathbf{a}, \mathbf{z}, \mathbf{B}, \mathbf{X})$	$(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Gamma}, \boldsymbol{\Xi})$

Table 3: The various forms of nabla.

Description		Glyph
Upright	Serif	∇
	Bold serif	$\boldsymbol{\nabla}$
	Bold sans	∇
Italic	Serif	∇
	Bold serif	$\boldsymbol{\nabla}$
	Bold sans	∇

4.3 Symbols requiring special attention

4.3.1 Nabla

The symbol ∇ comes in the six forms shown in table 3. We want an individual option to specify whether we want upright or italic nabla by default (when either upright or italic nabla is used in the source). \TeX classically uses an upright nabla, but iso standards differ (I think). The package options `nabla=upright` and `nabla=italic` switch between the two choices. This is then inherited through `\mathbf{f}`; `\mathit{f}` and `\mathup{f}` can be used to force one way or the other.

`nabla=italic` is implicit when using `math-style=ISO` and `nabla=upright` follows both `math-style=TeX` and `math-style=French`.

4.3.2 Partial

The same applies to the symbols U+2202: PARTIAL DIFFERENTIAL and U+1D715: MATH ITALIC PARTIAL DIFFERENTIAL.

At time of writing, both the Cambria Math and STIX fonts display these two glyphs in the same italic style, but this is hopefully a bug that will be corrected in the future — the ‘plain’ partial differential should really have an upright shape.

Use the `partial=upright` or `partial=italic` package options to specify which one you would like. The default is (always, unless someone requests and

Table 4: The various forms of the partial differential. Note that in the fonts used to display these glyphs, the first upright partial is incorrectly shown in an italic style.

Description		Glyph
Regular	Upright	∂
	Italic	∂
Bold	Upright	∂
	Italic	∂
Sans bold	Upright	∂
	Italic	∂

argues otherwise) `partial=italic`.¹

See table 4 for the variations on the partial differential symbol.

4.3.3 Epsilon and phi: ϵ vs. ε and ϕ vs. φ

\TeX defines `\epsilon` to look like ε and `\varepsilon` to look like ϵ . The Unicode glyph directly after delta and before zeta is ‘epsilon’ and looks like ϵ ; there is a subsequent variant of epsilon that looks like ε . This creates a problem. People who use unicode input won’t want their glyphs transforming; \TeX users will be confused that what they think as ‘normal epsilon’ is actual the ‘variant epsilon’. And the same problem exists for ‘phi’.

We have a package option to control this behaviour. With `\vargreek-shape=TeX`, `\phi` and `\epsilon` produce ϕ and ϵ and `\varphi` and `\varepsilon` produce φ and ε . With `\vargreek-shape=unicode`, these symbols are swapped. Note, however, that unicode characters are not affected by this option. That is, no remapping occurs of the characters/glyphs, only the control sequences.

Unless `math-style=literal` is in effect, the default is to use `\vargreek-shape=TeX`.

U+3B5: GREEK SMALL LETTER EPSILON

U+3F5: GREEK LUNATE EPSILON SYMBOL

U+3C6: GREEK SMALL LETTER PHI

U+3D5: GREEK SMALL LETTER SCRIPT PHI

¹A good argument would revolve around some international standards body recommending upright over italic. I just don’t have the time right now to look it up.

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
 $abcdefghijklmnopqrstuvwxyz$
 $AB\Gamma\Delta EZH\Theta\text{I}\text{K}\Lambda\text{M}\text{N}\Xi\text{O}\Pi\rho\sigma\tau\upsilon\phi\chi\psi\Omega$
 $\alpha\beta\gamma\delta\epsilon\zeta\eta\theta\vartheta\iota\kappa\lambda\mu\nu\xi\omicron\pi\rho\varsigma\sigma\tau\upsilon\phi\chi\psi\omega$

(a) Package option [math-style=ISO]

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
 $abcdefghijklmnopqrstuvwxyz$
 $AB\Gamma\Delta EZH\Theta\Box\text{I}\text{K}\Lambda\text{M}\text{N}\Xi\text{O}\Pi\rho\sigma\tau\upsilon\phi\chi\psi\Omega$
 $\alpha\beta\gamma\delta\epsilon\zeta\eta\theta\vartheta\iota\kappa\lambda\mu\nu\xi\omicron\pi\rho\varsigma\sigma\tau\upsilon\phi\chi\psi\omega$

(b) Package option [math-style=TeX]

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
 $abcdefghijklmnopqrstuvwxyz$
 $AB\Gamma\Delta EZH\Theta\Box\text{I}\text{K}\Lambda\text{M}\text{N}\Xi\text{O}\Pi\rho\sigma\tau\upsilon\phi\chi\psi\Omega$
 $\alpha\beta\gamma\delta\epsilon\Box\zeta\eta\theta\vartheta\iota\kappa\lambda\mu\nu\xi\omicron\pi\rho\varsigma\sigma\tau\upsilon\phi\chi\psi\omega$

(c) Package option [math-style=French]

Figure 3: Example maths output demonstrating the math-style package option.

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
 $abcdefghijklmnopqrstuvwxyz$
 $AB\Gamma\Delta EZH\Theta\text{I}\text{K}\Lambda\text{M}\text{N}\Xi\text{O}\Pi\rho\sigma\tau\upsilon\phi\chi\psi\Omega$
 $\alpha\beta\gamma\delta\epsilon\zeta\eta\theta\iota\kappa\lambda\mu\nu\xi\omicron\pi\rho\varsigma\sigma\tau\upsilon\phi\chi\psi\omega\epsilon\vartheta\kappa\phi\rho\omega$

(a) Package option [bold-style=ISO]

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
 $abcdefghijklmnopqrstuvwxyz$
 $AB\Gamma\Delta EZH\Theta\text{I}\text{K}\Lambda\text{M}\text{N}\Xi\text{O}\Pi\rho\sigma\tau\upsilon\phi\chi\psi\Omega$
 $\alpha\beta\gamma\delta\epsilon\zeta\eta\theta\iota\kappa\lambda\mu\nu\xi\omicron\pi\rho\varsigma\sigma\tau\upsilon\phi\chi\psi\omega\epsilon\vartheta\kappa\phi\rho\omega$

(b) Package option [bold-style=TeX]

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
 $abcdefghijklmnopqrstuvwxyz$
 $AB\Gamma\Delta EZH\Theta\text{I}\text{K}\Lambda\text{M}\text{N}\Xi\text{O}\Pi\rho\sigma\tau\upsilon\phi\chi\psi\Omega$
 $\alpha\beta\gamma\delta\epsilon\zeta\eta\theta\iota\kappa\lambda\mu\nu\xi\omicron\pi\rho\varsigma\sigma\tau\upsilon\phi\chi\psi\omega\epsilon\vartheta\kappa\phi\rho\omega$

(c) Package option [bold-style=French]

Figure 4: Example maths output demonstrating the bold-style package option.

File I

The unicode-math package

This is the package.

```
1 \ProvidesPackage{unicode-math}  
2 [2009/09/17 v0.4 Unicode maths in XeLaTeX]
```

5 Things we need

Packages

```
3 \RequirePackage{expl3}[2009/08/12]  
4 \RequirePackage{xparse}[2009/08/31]  
5 \RequirePackage{fontspec}
```

Start using L^AT_EX3 — finally!

```
6 \ExplSyntaxOn
```

Package wrangling:

- Since the mathcode of `\-` is greater than eight bits, this piece of `\AtBeginDocument` code from `amsmath` dies if we try and set the maths font in the preamble:

```
7 \ifpackageloaded{amsmath}{  
8   \tl_remove_in:Nn \@begindocumenthook {  
9     \mathchardef\std@minus\mathcode'\-\relax  
10    \mathchardef\std@equal\mathcode'\=\relax  
11   }  
12 }{}
```

Counters and conditionals

```
13 \newcounter{um@fam}  
14 \newif\if@um@fontspec@feature  
15 \newif\if@um@ot@math@
```

For math-style:

```
16 \newif\if@um@literal  
17 \newif\if@um@upGreek  
18 \newif\if@um@upgreek  
19 \newif\if@um@upLatin  
20 \newif\if@um@uplatin
```

For bold-style:

```
21 \newif\if@um@bfliteral  
22 \newif\if@um@bfupGreek  
23 \newif\if@um@bfupgreek
```

```

24 \newif\if@um@bfupLatin
25 \newif\if@um@bfuplatin

```

For nabla:

```

26 \newif\if@um@upNabla
27 \newif\if@um@uppartial
28 \bool_new:N \g_um_texgreek_bool

```

5.0.4 Alphabet unicode positions

Before we begin, let's define the positions of the various unicode alphabets so that our code is a little more readable.²

```

29 \def\um@usv@num{'\0}
30 \def\um@usv@upLatin{'\A}
31 \def\um@usv@uplatin{'\a}
32 \def\um@usv@upGreek{"391}
33 \def\um@usv@upgreek{"3B1}
34 \def\um@usv@itLatin{"1D434}
35 \def\um@usv@itlatin{"1D44E}
36 \def\um@usv@itGreek{"1D6E2}
37 \def\um@usv@itgreek{"1D6FC}
38 \def\um@usv@bbnum{"1D7D8}
39 \def\um@usv@bbLatin{"1D538}
40 \def\um@usv@bblatin{"1D552}
41 \def\um@usv@scrLatin{"1D49C}
42 \def\um@usv@scrlatin{"1D4B6}
43 \def\um@usv@frakLatin{"1D504}
44 \def\um@usv@fraklatin{"1D51E}
45 \def\um@usv@sfnum{"1D7E2}
46 \def\um@usv@sfupLatin{"1D5A0}
47 \def\um@usv@sfuplatin{"1D5BA}
48 \def\um@usv@sflatin {"1D5BA}
49 \def\um@usv@sfitLatin{"1D608}
50 \def\um@usv@sfitlatin{"1D622}
51 \def\um@usv@ttnum{"1D7F6}
52 \def\um@usv@ttLatin{"1D670}
53 \def\um@usv@ttlLatin{"1D68A}

```

Bold:

```

54 \def\um@usv@bfnum{"1D7CE}
55 \def\um@usv@bfupLatin{"1D400}
56 \def\um@usv@bfuplatin{"1D41A}
57 \def\um@usv@bflatin {"1D41A}
58 \def\um@usv@bfupGreek{"1D6A8}
59 \def\um@usv@bfupgreek{"1D6C2}

```

²'u.s.v.' stands for 'unicode scalar value'.

```

60 \def\um@usv@bfitLatin{"1D468}
61 \def\um@usv@bfitlatin{"1D482}
62 \def\um@usv@bfitGreek{"1D71C}
63 \def\um@usv@bfitgreek{"1D736}
64 \def\um@usv@bffrakLatin{"1D56C}
65 \def\um@usv@bffraklatin{"1D586}
66 \def\um@usv@bfscrLatin{"1D4D0}
67 \def\um@usv@bfscrlatin{"1D4EA}
68 \def\um@usv@bfsfnum{"1D7EC}
69 \def\um@usv@bfsfupLatin{"1D5D4}
70 \def\um@usv@bfsfuplatin{"1D5EE}
71 \def\um@usv@bfsflatin {"1D5EE}
72 \def\um@usv@bfsfupGreek{"1D756}
73 \def\um@usv@bfsfupgreek{"1D770}
74 \def\um@usv@bfsfitLatin{"1D63C}
75 \def\um@usv@bfsfitlatin{"1D656}
76 \def\um@usv@bfsfitGreek{"1D790}
77 \def\um@usv@bfsfitgreek{"1D7AA}

```

Greek variants:

```

78 \def\um@usv@varTheta{"3F4}
79 \def\um@usv@Digamma{"3DC}
80 \def\um@usv@varepsilon{"3F5}
81 \def\um@usv@vartheta{"3D1}
82 \def\um@usv@varkappa{"3F0}
83 \def\um@usv@varphi{"3D5}
84 \def\um@usv@varrho{"3F1}
85 \def\um@usv@varpi{"3D6}
86 \def\um@usv@digamma{"3DD}

```

Bold:

```

87 \def\um@usv@bfvarTheta{"1D6B9}
88 \def\um@usv@bfDigamma{"1D7CA}
89 \def\um@usv@bfvarepsilon{"1D6DC}
90 \def\um@usv@bfvartheta{"1D6DD}
91 \def\um@usv@bfvarkappa{"1D6DE}
92 \def\um@usv@bfvarphi{"1D6DF}
93 \def\um@usv@bfvarrho{"1D6E0}
94 \def\um@usv@bfvarpi{"1D6E1}
95 \def\um@usv@bfdigamma{"1D7CB}

```

Italic Greek variants:

```

96 \def\um@usv@ith{"210E}
97 \def\um@usv@itvarTheta{"1D6F3}
98 \def\um@usv@itvarepsilon{"1D716}
99 \def\um@usv@itvartheta{"1D717}
100 \def\um@usv@itvarkappa{"1D718}
101 \def\um@usv@itvarphi{"1D719}

```

```

102 \def\um@usv@itvarrho{"1D71A}
103 \def\um@usv@itvarpi{"1D71B}

```

Bold:

```

104 \def\um@usv@bfuph{"1D421}
105 \def\um@usv@bfith{"1D489}
106 \def\um@usv@bfitvarTheta{"1D72D}
107 \def\um@usv@bfitvarepsilon{"1D750}
108 \def\um@usv@bfitvartheta{"1D751}
109 \def\um@usv@bfitvarkappa{"1D752}
110 \def\um@usv@bfitvarphi{"1D753}
111 \def\um@usv@bfitvarrho{"1D754}
112 \def\um@usv@bfitvarpi{"1D755}

```

Nabla:

```

113 \def\um@usv@Nabla{"2207}
114 \def\um@usv@itNabla{"1D6FB}
115 \def\um@usv@bfNabla{"1D6C1}
116 \def\um@usv@bfitNabla{"1D735}
117 \def\um@usv@bfsfNabla{"1D76F}
118 \def\um@usv@bfsfitNabla{"1D7A9}

```

Partial:

```

119 \def\um@usv@partial{"2202}
120 \def\um@usv@itpartial{"1D715}
121 \def\um@usv@bfpartial{"1D6DB}
122 \def\um@usv@bfitpartial{"1D74F}
123 \def\um@usv@bfsfpartial{"1D789}
124 \def\um@usv@bfsfitpartial{"1D7C3}

```

5.1 Package options

xkeyval's package support is used here.

math-style

```

125 \define@choicekey*{unicode-math.sty}
126   {math-style}[\@tempa\@tempb]{iso,tex,french,literal}{
127   \ifcase\@tempb\relax
128     \um@upGreekfalse
129     \um@upgreekfalse
130     \um@upLatinfalse
131     \um@uplatinfalse
132     \um@bfupGreekfalse
133     \um@bfupgreekfalse
134     \um@uppartialfalse
135     \um@bfupLatinfalse
136     \um@bfuplatinfalse

```

```

137 \um@upNablafalse
138 \bool_set_false:N \g_um_texgreek_bool
139 \or
140 \um@upGreektrue
141 \um@upgreekfalse
142 \um@upLatinfalse
143 \um@uplatinfalse
144 \um@bfupGreektrue
145 \um@bfupgreekfalse
146 \um@uppartialfalse
147 \um@bfupLatintrue
148 \um@bfuplatintrue
149 \um@upNablatrue
150 \bool_set_true:N \g_um_texgreek_bool
151 \or
152 \um@upGreektrue
153 \um@upgreektrue
154 \um@upLatintrue
155 \um@uplatinfalse
156 \um@bfupGreektrue
157 \um@bfupgreektrue
158 \um@uppartialtrue
159 \um@bfupLatintrue
160 \um@bfuplatintrue
161 \um@upNablatrue
162 \bool_set_false:N \g_um_texgreek_bool
163 \or
164 \um@literaltrue
165 \um@bfliteraltrue
166 \bool_set_false:N \g_um_texgreek_bool
167 \fi
168 }

```

bold-style

```

169 \define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,french,literal}{
170 \ifcase\@tempb\relax
171 \um@bfupGreekfalse
172 \um@bfupgreekfalse
173 \um@uppartialfalse
174 \um@bfupLatinfalse
175 \um@bfuplatinfalse
176 \or
177 \um@bfupGreektrue
178 \um@bfupgreekfalse
179 \um@uppartialfalse
180 \um@bfupLatintrue

```

```

181 \um@bfuplatintrue
182 \or
183 \um@bfupGreektrue
184 \um@bfupgreektrue
185 \um@uppartialtrue
186 \um@bfupLatintrue
187 \um@bfuplatintrue
188 \or
189 \um@bfliteraltrue
190 \fi
191 }

```

Symbol obliqueness

```

192 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
193 \ifcase\@tempb\relax
194 \um@upNablatrue
195 \or
196 \um@upNablafalse
197 \fi
198 }
199 \cs_set:Nn \um_setup_nabla: {
200 \if@um@upNabla
201 \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@Nabla }
202 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfNabla }
203 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfNabla }
204 \else
205 \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@itNabla }
206 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfitNabla }
207 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfitNabla }
208 \fi
209 }
210 \define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
211 \ifcase\@tempb\relax
212 \um@uppartialtrue
213 \or
214 \um@uppartialfalse
215 \fi
216 }
217 \cs_set:Nn \um_setup_partial: {
218 \if@um@uppartial
219 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@partial }
220 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfpartial }
221 \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfpartial }
222 \else
223 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@itpartial }
224 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfitpartial }

```

```

225     \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfitpartial }
226     \fi
227 }

```

Epsilon and phi shapes

```

228 \define@choicekey*{unicode-math.sty}{vargreek-shape}[\@tempa\@tempb]{unicode,TeX}{
229     \ifcase\@tempb\relax
230         \bool_set_false:N \g_um_texgreek_bool
231     \or
232         \bool_set_true:N \g_um_texgreek_bool
233     \fi
234 }

235 \ExecuteOptionsX{math-style=TeX}
236 \ProcessOptionsX

```

5.2 Overcoming \@onlypreamble

The requirement of only setting up the maths fonts is now removed. The following list might be overly ambitious.

```

237 \tl_map_inline:nn {
238     \new@mathgroup
239     \cdp@list
240     \cdp@elt
241     \DeclareMathSizes
242     \@DeclareMathSizes
243     \newmathalphabet
244     \newmathalphabet@@
245     \newmathalphabet@@@
246     \DeclareMathVersion
247     \define@mathalphabet
248     \define@mathgroup
249     \addtoversion
250     \version@list
251     \version@elt
252     \alpha@list
253     \alpha@elt
254     \restore@mathversion
255     \init@restore@version
256     \dorestore@version
257     \process@table
258     \new@mathversion
259     \DeclareSymbolFont
260     \group@list
261     \group@elt
262     \new@symbolfont

```



```

263 \SetSymbolFont
264 \SetSymbolFont@
265 \get@cdp
266 \DeclareMathAlphabet
267 \new@mathalphabet
268 \SetMathAlphabet
269 \SetMathAlphabet@
270 \DeclareMathAccent
271 \set@mathaccent
272 \DeclareMathSymbol
273 \set@mathchar
274 \set@mathsymbol
275 \DeclareMathDelimiter
276 \@xxDeclareMathDelimiter
277 \@DeclareMathDelimiter
278 \@xDeclareMathDelimiter
279 \set@mathdelimiter
280 \set@@mathdelimiter
281 \DeclareMathRadical
282 \mathchar@type
283 \DeclareSymbolFontAlphabet
284 \DeclareSymbolFontAlphabet@
285 }{
286 \tl_remove_in:Nn \@preamblecmds {\do#1}
287 }

```

5.3 Other things

`\um@fontdimen@percent` #1 : Font dimen number
`\fontdimens` 10, 11, and 65 aren't actually dimensions, they're percentage values given in units of sp. This macro takes a font dimension number and outputs the decimal value of the associated parameter.

0.73	<code>\font\tmpfont="Cambria Math"</code>
0.60	<code>\um@fontdimen@percent{10}{\tmpfont}\</code>
0.65	<code>\um@fontdimen@percent{11}{\tmpfont}\</code>
	<code>\um@fontdimen@percent{65}{\tmpfont}</code>

```

288 \def\um@fontdimen@percent#1#2{
289   0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax
290 }

```

`\um@scaled@apply` #1 : A math style
 #2 : Macro that takes a non-delimited length argument (like `\kern`)
 #3 : Length control sequence to be scaled according to the math style

This macro is used to scale the lengths reported by `\fontdimen` according to the scale factor for script- and scriptscript-size objects.

```

291 \def\um@scaled@apply#1#2#3{
292   \ifx#1\scriptstyle
293     #2\um@fontdimen@percent{10}\um@font#3
294   \else
295     \ifx#1\scriptscriptstyle
296       #2\um@fontdimen@percent{11}\um@font#3
297     \else
298       #2#3%
299     \fi
300   \fi
301 }
```

6 Fundamentals

6.1 Enlarging the number of maths families

To start with, we've got a power of two as many `\fams` as before. So (from `ltfssbas.dtx`) we want to redefine

```

302 \def\new@mathgroup{\alloc@8\mathgroup\chardef\@cclvi}
303 \let\newfam\new@mathgroup
```

This is sufficient for L^AT_EX's `\DeclareSymbolFont`-type commands to be able to define 256 named maths fonts. Now we need a new `\DeclareMathSymbol`.

6.2 `\DeclareMathSymbol` for unicode ranges

This command is a bit funny at the moment; it doesn't define the actual macro for almost all of the symbols passed to it, but it does assign the `\XeTeXmathchar`.

```

\um@mathsymbol #1 : Symbol, e.g., \alpha
                #2 : Type, e.g., \mathalpha
                #3 : Math font name, e.g., operators
                #4 : Slot, e.g., "221E
304 \def \um@mathsymbol#1#2#3#4{
305   \expandafter\um@set@mathsymbol\csname sym#3\endcsname#1#2{#4}}
```

The final macros that actually define the maths symbol with X_ET_EX primitives.

```

\um@set@mathsymbol #1 : Symbol font number
                   #2 : Symbol macro, e.g., \alpha
                   #3 : Type, e.g., \mathalpha
                   #4 : Slot, e.g., "221E
```

If the symbol definition is for a macro. There are a bunch of tests to perform to process the various characters.

```
306 \def\um@set@mathsymbol#1#2#3#4{
```

Operators In the examples following, say we’re defining for the symbol \sum (Σ).

```
307 \ifx\mathop#3\relax
```

In order for literal unicode characters to be used in the source and still have the correct limits behaviour, big operators are made math-active.

The active math char is \let to the macro $\sum@op$.

```
308 \begingroup
309 \char_make_active:n {#4}
310 \global\mathcode#4="8000\relax
311 \um@scanactivedef #4 \@nil { \csname\string#2@op\endcsname }
312 \endgroup
```

Some of these require a \nolimits suffix. This is controlled by the $\um@nolimits$ macro, which contains a list of such characters. This list is checked dynamically because we’re not interested in efficiency. Or something. This allows the list to be updated in the middle of a document.

Declare the plain old \mathchardef for the control sequence $\sum@sym$.

```
313 \expandafter\global\expandafter\XeTeXmathchardef
314 \csname\string#2@sym\endcsname
315 =" \mathchar@type#3 #1 #4\relax
```

Now define $\sum@op$ as $\sum@sym$, followed by \nolimits if necessary.

```
316 \cs_gset:cpn { \string#2 @op } {
317 \csname\string#2@sym\endcsname
318 \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
319 \ifin@
320 \expandafter\nolimits
321 \fi
322 }
```

Don’t forget that the actual \sum macro is simply defined in terms of the literal unicode symbol!

```
323 \else
```

Radicals Needs to be before the delimiters because the radical is, for some reason, \mathopen .

```
324 \expandafter\in@\expandafter#2\expandafter{\um@radicals,}
325 \ifin@
326 \cs_gset:cpn {\cs_to_str:N #2 sign} { \XeTeXradical #1 #4 \relax }
327 \else
```

Delimiters TODO: sort out which of these three declarations are necessary! (Definitely the first, to work with `\left/\right`.)

```

328     \ifx\mathopen#3\relax
329         \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
330         \global\XeTeXdelcode#4=#1 #4\relax
331         \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
332     \else
333         \ifx\mathclose#3\relax
334             \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
335             \global\XeTeXdelcode#4=#1 #4\relax
336             \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
337         \else

```

Accents

```

338         \ifx\mathaccent#3\relax
339             \cs_gset:Npx #2 {\XeTeXmathaccent "\mathchar@type#3 #1 #4\relax}
340         \else

```

And finally, the general case. We define the unicode mathcode for the character. The macro is defined generically in terms of the unicode character.

```

341             \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
342         \fi
343     \fi
344 \fi
345 \fi
346 \fi
347 }

```

`\um_set_mathcode:nnnn` [For later] or if it's for a character code (just a wrapper around the primitive). Note that this declaration *isn't* global so that it can be constrained by grouping inside math alphabet switches.

```

348 \cs_set:Nn \um_set_mathcode:nnnn {
349     \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
350 }

```

6.3 The main `\setmathfont` macro

Here's the simplest usage:

$$Ax \triangleq \nabla \times \mathcal{Z}$$

```

\setmathfont{Asana Math}
$Ax \eqdef \nabla \times \mathscr{Z}

```

An interesting (perhaps useless) example of the Range feature:

$$F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st} f(t) dt$$

```

\setmathfont[Colour=000000]{Asana Math}
\setmathfont[Range={\mathop}, Colour=FF0000]{Asana Math}
\setmathfont[Range={\equal}, Colour=009900]{Asana Math}
\setmathfont[Range={\mathopen,\mathclose},
              Colour=0000FF]{Asana Math}
\l[
F(s)=\mscrL\{f(t)\}=\int_0^\infty \mathup{e}^{-st}f(t)\,\mathup{d} t
\l]

```

Using a Range including large character sets such as `\mathrel`, `\mathalpha`, *etc.*, is *very slow*! I hope to improve the performance somehow.

`\setmathfont` [#1]: font features

#2 : font name

```
351 \DeclareDocumentCommand \setmathfont { 0{ } m } {
```

- Erase any conception L^AT_EX has of previously defined math symbol fonts; this allows `\DeclareSymbolFont` at any point in the document.

```
352 \let\glb@currsizel\relax
```

- To start with, assume we're defining the font for every math symbol character.

```
353 \let\um@char@range\@empty
```

```
354 \let\um@char@num@range\@empty
```

- Tell fontspec that maths font features are actually allowed.

```
355 \um@fontspec@featuretrue
```

- Grab the current size information (is this robust enough? Maybe it should be preceded by `\normalsize`).

```
356 \csname S@f@size\endcsname
```

- Set the name of the math version being defined. (obviously more needs to be done here!)

```
357 \def\um@mversion{normal}
```

```
358 \DeclareMathVersion{\um@mversion}
```

Define default font features for the script and scriptscript font. (This needs to be generalised so users can override it.)

```
359 \tl_set:Nn \l_um_script_features_tl {ScriptStyle}
```

```
360 \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
```

```
361 \tl_set:Nn \l_um_script_font_tl {#2}
```

```
362 \tl_set:Nn \l_um_sscript_font_tl {#2}
```

Use `fontspec` to select a font to use. The macro `\Set $\langle size \rangle$` contains the definitions of the sizes used for maths letters, subscripts and subsubscripts in `\tf@size`, `\sf@size`, and `\ssf@size`, respectively.

```

363 \setkeys*{um}[options]{#1}
364 \edef\@tempa{\noexpand\zf@fontspec{
365     Script = Math,
366     SizeFeatures = {
367         {Size = \tf@size-} ,
368         {Size = \sf@size-\tf@size ,
369         Font = \l_um_script_font_tl ,
370         \l_um_script_features_tl
371         } ,
372         {Size = -\sf@size ,
373         Font = \l_um_sscript_font_tl ,
374         \l_um_sscript_features_tl
375         }
376     },
377     \XKV@rm
378 }{#2}
379 }
380 \@tempa

```

Probably want to check there that we’re not creating multiple symbol fonts with the same NFSS declaration.

Check for the correct number of `\fontdimens`:

```

381 \font\um@font="#2"\relax
382 \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
383 \um@ot@math@true
384 \else
385 \PackageWarningNoLine{unicode-math}{
386     The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
387     Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
388     in~ a~ substandard~ manner
389 }
390 \fi

```

If we’re defining the full unicode math repertoire, then we skip all the parsing processing needed if we’re only defining a subset.

- Math symbols are defined with `\UnicodeMathSymbol`; see section §6.3.1 for the individual definitions

```

391 \ifx\um@char@range\@empty
392 \tl_set:Nn \um_symfont_tl {um@allsym}
393 \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
394 \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
395 \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
396 \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn

```

```

397 \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
398 \else
399 \stepcounter{um@fam}
400 \tl_set:Nx \um_symfont_tl {um@fam\theum@fam}
401 \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn
402 \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
403 \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
404 \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
405 \fi

```

Now defined `\um_symfont_tl` as the L^AT_EX math font to access everything:

```

406 \DeclareSymbolFont{\um_symfont_tl}
407 {\encodingdefault}{\zf@family}{\mddefault}{\updefault}

```

And now we input every single maths char. See File II for the source to `unicode-math.tex` which is used to create `unicode-math-table.tex`.

```

408 \@input{unicode-math-table.tex}

```

Finally,

- Set up shapes for italic/upright or ordinary/var symbols as per package options.
- Remap symbols that don't take their natural mathcode
- Activate any symbols that need to be math-active
- Setup all symbols not covered by the table (mostly alphanumerics)
- Setup the maths alphabets (`\mathbf` etc.)

```

409 \um_setup_shapes:
410 \um_remap_symbols:
411 \um_setup_mathactives:
412 \um_setup_alphanum:
413 \um_setup_alphabets:

```

End of the `\setmathfont` macro.

```

414 }

415 \cs_new:Nn \um_setup_shapes: {
416 \um_setup_nabla:
417 \um_setup_partial:
418 }

```

6.3.1 Functions for setting up symbols with mathcodes

`\um_process_symbol_noparse:nnnn` If the Range font feature has been used, then only a subset of the unicode glyphs are to be defined. See section §7.3 for the code that enables this.

```

419 \cs_set:Nn \um_process_symbol_noparse:nnnn {
420   \um@mathsymbol{#2}{#3}{\um_symfont_tl}{#1}
421 }
422 \cs_set:Nn \um_process_symbol_parse:nnnn {
423   \um@parse@term{#1}{#2}{#3}{
424     \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
425   }
426 }

```

`\um_remap_symbols:` This function is used to define the mathcodes for those chars which should be mapped to a different glyph than themselves.

```

\um_remap_symbol_noparse:nnn
\um_remap_symbol_parse:nnn
427 \cs_new:Nn \um_remap_symbols: {
428   \um_remap_symbol:nnn{"2D}{\mathbin}{"02212}% hyphen to minus
429   \if@um@literal
430     \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
431     \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
432     \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
433     \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
434   \else
435     \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um_Nabla_up_or_it_usv}
436     \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um_partial_up_or_it_usv}
437   \fi

```

Some of these in the `bfliteral` block may be redundant, but that's okay:

```

438   \if@um@bfliteral
439     \um_remap_symbol:nnn {\um@usv@bfNabla}{\mathord}{\um@usv@bfNabla}
440     \um_remap_symbol:nnn {\um@usv@bfitNabla}{\mathord}{\um@usv@bfitNabla}
441     \um_remap_symbol:nnn {\um@usv@bfsfNabla}{\mathord}{\um@usv@bfsfNabla}
442     \um_remap_symbol:nnn {\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsfitNabla}
443     \um_remap_symbol:nnn {\um@usv@bfpartial}{\mathord}{\um@usv@bfpartial}
444     \um_remap_symbol:nnn {\um@usv@bfitpartial}{\mathord}{\um@usv@bfitpartial}
445     \um_remap_symbol:nnn {\um@usv@bfsfpartial}{\mathord}{\um@usv@bfsfpartial}
446     \um_remap_symbol:nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
447   \else
448     \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla}{\mathord}{\um_bfNabla_up_or_it_usv}
449     \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla}{\mathord}{\um_bfsfNabla_up_or_it_usv}
450     \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um_bfpartial_up_or_it_usv}
451     \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um_bfsfpartial_up_or_it_usv}
452   \fi
453 }

```

Where `\um_remap_symbol:nnn` is defined to be one of these two, depending on the range setup:

```

454 \cs_new:Nn \um_remap_symbol_parse:nnn {
455   \um@parse@term {#3} {\@nil} {#2} {
456     \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
457   }

```



```

458 }
459 \cs_new:Nn \um_remap_symbol_noparse:nnn {
460   \clist_map_inline:nn {#1} {
461     \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
462   }
463 }

```

6.3.2 Active math characters

There are more math active chars later in the subscript/superscript section. But they don't need to be able to be typeset directly.

`\um_setup_mathactives:`

```

464 \cs_new:Nn \um_setup_mathactives: {
465   \um_make_mathactive:nNN {"2032} \primesingle \mathord
466 }

```

`\um_make_mathactive:nNN` Makes #1 a mathactive char, and gives cs #2 the meaning of mathchar #1 with class #3. You are responsible for giving active #1 a particular meaning!

```

467 \cs_new:Nn \um_make_mathactive:nNN {
468   \XeTeXmathchardef #2 = "\mathchar@type #3
469                               \csname sym\um_symfont_tl\endcsname
470                               #1 \scan_stop:
471   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:
472 }

```

6.3.3 Maths alphabets' character mapping

We want it to be convenient for users to actually type in maths. The ASCII Latin characters should be used for italic maths, and the text Greek characters should be used for upright/italic (depending on preference) Greek, if desired.

`\um_setup_alphanum:` All symbols input that aren't defined directly in unicode-math-table.

```

473 \cs_set:Nn \um_setup_alphanum: {
474   \ifx\um@char@range\@empty
475     \um_map_chars_numbers:nn {\um@usv@num}{\um@usv@num}

```

Normal weight

```

476   \if@um@literal
477     \um_setup_literals:
478   \else
479     \um_setup_Latin:
480     \um_setup_latin:
481     \um_setup_Greek:
482     \um_setup_greek:
483   \fi

```

Bold

```
484 \if@um@bfliteral
485 \um_setup_bf_literals:
486 \else
487 \if@um@bfuplatin
488 \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfupLatin}
489 \else
490 \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
491 \fi
492 \if@um@bfuplatin
493 \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfuplatin}
494 \else
495 \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
496 \fi
497 \if@um@bfupgreek
498 \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfupGreek}
499 \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
500 \else
501 \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}
502 \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
503 \fi
504 \if@um@bfupgreek
505 \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
506 \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
507 \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
508 \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
509 \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
510 \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
511 \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
512 \else
513 \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
514 \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
515 \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
516 \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
517 \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
518 \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
519 \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
520 \fi
521 \fi
522 \else
: TODO : what is supposed to happen here?
523 \fi
524 }
```

6.3.4 Functions for setting up the maths alphabets

`\um_mathmap_noparse:Nnn` #1 : Maths alphabet, *e.g.*, `\mathbb`
 #2 : Input slot(s), *e.g.*, the slot for ‘A’ (comma separated)
 #3 : Output slot, *e.g.*, the slot for ‘A’
 Adds `\um_set_mathcode:nnnn` declarations to the specified maths alphabet’s definition (*e.g.*, `\um@mathscr`). Uses `\um@addto@mathmap` (below) to expand the name of the current symbol font.

```

525 \cs_set:Nn \um_mathmap_noparse:Nnn {
526   \clist_map_inline:nn {#2} {
527     \exp_args:No \um@addto@mathmap \um_symfont_tl {##1}{#1}{#3}
528   }
529 }
```

`\um_mathmap_parse:Nnn` #1 : Maths alphabet, *e.g.*, `\mathbb`
 #2 : Input slot(s), *e.g.*, the slot for ‘A’ (comma separated)
 #3 : Output slot, *e.g.*, the slot for ‘A’
 When `\um@parse@term` is executed, it populates the `\um@char@num@range` macro with slot numbers corresponding to the specified range. This range is used to conditionally add `\um_set_mathcode:nnnn` declarations to the maths alphabet definition (*e.g.*, `\um@mathscr`).

```

530 \cs_set:Nn \um_mathmap_parse:Nnn {
531   \clist_map_inline:Nn \um@char@num@range {
532     \ifnum##1=##3\relax
533       \clist_map_inline:nn {#2} {
534         \exp_args:No \um@addto@mathmap \um_symfont_tl {####1}{#1}{#3}
535       }
536     \fi
537   }
538 }
```

`\um@addto@mathmap` #1 : Math symbol font, always/usually the expansion of `\um_symfont_tl`
 #2 : Input slot, *e.g.*, the slot for ‘A’
 #3 : Maths alphabet, *e.g.*, `\mathbb`
 #4 : Output slot, *e.g.*, the slot for ‘A’
 This macro is used so that `\um_symfont_tl` can be expanded before entering the `\g@addto@macro` command.

```

539 \newcommand\um@addto@mathmap[4]{
540   \expandafter\g@addto@macro
541     \csname um_setup_\cs_to_str:N #3:\endcsname{
542     \um_set_mathcode:nnnn{#2}{\mathalpha}{#1}{#4}
543   }
544 }
```


























6.4 (Big) operators


Turns out that \LaTeX is clever enough to deal with big operators for us automatically with `\XeTeXmathchardef`. Amazing!


However, the limits aren't set automatically; that is, we want to define, a *la Plain \TeX* `etc.`, `\def\int{\intop\nolimits}`, so there needs to be a transformation from `\int` to `\intop` during the expansion of `\UnicodeMathSymbol` in the appropriate contexts.

Following is a table of every math operator (`\mathop`) defined in `unicode-math-table.tex`, from which a subset need to be flagged for `\nolimits` adjustments. The limits behaviour as specified by `unicode-math` are shown (with grey 'scripts').

USV	Ex.	Macro	Description
U+02140	$\sum\limits_0^1$	<code>\Bbbsum</code>	DOUBLE-STRUCK N-ARY SUMMATION
U+0220F	$\prod\limits_0^1$	<code>\prod</code>	PRODUCT OPERATOR
U+02210	$\coprod\limits_0^1$	<code>\coprod</code>	COPRODUCT OPERATOR
U+02211	$\sum\limits_0^1$	<code>\sum</code>	SUMMATION OPERATOR
U+0222B	$\int\limits_0^1$	<code>\int</code>	INTEGRAL OPERATOR
U+0222C	$\iint\limits_0^1$	<code>\iint</code>	DOUBLE INTEGRAL OPERATOR
U+0222D	$\iiint\limits_0^1$	<code>\iiint</code>	TRIPLE INTEGRAL OPERATOR
U+0222E	$\oint\limits_0^1$	<code>\oint</code>	CONTOUR INTEGRAL OPERATOR
U+0222F	$\oiint\limits_0^1$	<code>\oiint</code>	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230	$\oiiint\limits_0^1$	<code>\oiiint</code>	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231	$\int\limits_0^1$	<code>\intclockwise</code>	CLOCKWISE INTEGRAL
U+02232	$\oint\limits_0^1$	<code>\varointclockwise</code>	CONTOUR INTEGRAL, CLOCKWISE
U+02233	$\oint\limits_0^1$	<code>\ointctrackwise</code>	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0	$\bigwedge\limits_0^1$	<code>\bigwedge</code>	LOGICAL OR OPERATOR
U+022C1	$\bigvee\limits_0^1$	<code>\bigvee</code>	LOGICAL AND OPERATOR
U+022C2	$\bigcap\limits_0^1$	<code>\bigcap</code>	INTERSECTION OPERATOR
U+022C3	$\bigcup\limits_0^1$	<code>\bigcup</code>	UNION OPERATOR
U+027D5	$\left\lrcorner\limits_0^1$	<code>\leftouterjoin</code>	LEFT OUTER JOIN
U+027D6	$\right\lrcorner\limits_0^1$	<code>\rightouterjoin</code>	RIGHT OUTER JOIN

U+027D7		\fullouterjoin	FULL OUTER JOIN
U+027D8		\bigbot	LARGE UP TACK
U+027D9		\bigtop	LARGE DOWN TACK
U+029F8		\xsol	BIG SOLIDUS
U+029F9		\xbsol	BIG REVERSE SOLIDUS
U+02A00		\bigodot	N-ARY CIRCLED DOT OPERATOR
U+02A01		\bigoplus	N-ARY CIRCLED PLUS OPERATOR
U+02A02		\bigotimes	N-ARY CIRCLED TIMES OPERATOR
U+02A03		\bigcupdot	N-ARY UNION OPERATOR WITH DOT
U+02A04		\biguplus	N-ARY UNION OPERATOR WITH PLUS
U+02A05		\bigsqcap	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06		\bigsqcup	N-ARY SQUARE UNION OPERATOR
U+02A07		\conjquant	TWO LOGICAL AND OPERATOR
U+02A08		\disjquant	TWO LOGICAL OR OPERATOR
U+02A09		\bigtimes	N-ARY TIMES OPERATOR
U+02A0B		\sumint	SUMMATION WITH INTEGRAL
U+02A0C		\iiint	QUADRUPLE INTEGRAL OPERATOR
U+02A0D		\intbar	FINITE PART INTEGRAL
U+02A0E		\intBar	INTEGRAL WITH DOUBLE STROKE
U+02A0F		\fint	INTEGRAL AVERAGE WITH SLASH
U+02A10		\cirfnint	CIRCULATION FUNCTION
U+02A11		\awint	ANTICLOCKWISE INTEGRATION LINE INTEGRATION WITH RECTANGULAR
U+02A12		\rppoint	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13		\scpoint	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE
U+02A14		\npoint	POLE

U+02A15		<code>\pointint</code>	INTEGRAL AROUND A POINT OPERATOR
U+02A16		<code>\sqint</code>	QUATERNION INTEGRAL OPERATOR
U+02A17		<code>\intlarhk</code>	HOOK
U+02A18		<code>\intx</code>	INTEGRAL WITH TIMES SIGN
U+02A19		<code>\intcap</code>	INTEGRAL WITH INTERSECTION
U+02A1A		<code>\intcup</code>	INTEGRAL WITH UNION
U+02A1B		<code>\upint</code>	INTEGRAL WITH OVERBAR
U+02A1C		<code>\lowint</code>	INTEGRAL WITH UNDERBAR
U+02A1D		<code>\Join</code>	JOIN
U+02A1E		<code>\bigtriangleleft</code>	LARGE LEFT TRIANGLE OPERATOR
U+02A1F		<code>\zcmp</code>	Z NOTATION SCHEMA COMPOSITION
U+02A20		<code>\zpipe</code>	Z NOTATION SCHEMA PIPING
U+02A21		<code>\zproject</code>	Z NOTATION SCHEMA PROJECTION
U+02AFC		<code>\biginterleave</code>	LARGE TRIPLE VERTICAL BAR OPERATOR
U+02AFF		<code>\bigtalloblong</code>	N-ARY WHITE VERTICAL BAR

`\um@nolimits` This macro is a sequence containing those maths operators that require a `\nolimits` suffix. This list is used when processing `unicode-math-table.tex` to define such commands automatically (see the macro `\um@set@mathsymbol` on page 18). I’ve chosen essentially just the operators that look like integrals; hopefully a better mathematician can help me out here. I’ve a feeling that it’s more useful *not* to include the multiple integrals such as , but that might be a matter of preference.

```

545 \def\um@nolimits{
546   \@elt\int\@elt\iint\@elt\iiint\@elt\iiint\@elt\oint\@elt\oiint\@elt\oiint
547   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
548   \@elt\intbar\@elt\intBar\@elt\oint\@elt\cirfnint\@elt\awint\@elt\rppoint
549   \@elt\scpolint\@elt\ntopolint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
550   \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
551 }

```

`\addnolimits` This macro appends material to the macro containing the list of operators that don’t take limits. See example following for usage. Note at present that this command must have taken effect before `\setmathfont`.

```

552 \newcommand\addnolimits[1]{
553   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
554 }

```

`\removenolimits` Can this macro be given a better name? It removes (globally) an item from the `nolimits` list. See example following for usage.

```

555 \def\removenolimits#1{
556   \begingroup
557     \def\@elt##1{
558       \ifx##1#1\else
559         \noexpand\@elt\noexpand##1
560       \fi}
561     \xdef\um@nolimits{\um@nolimits}
562   \endgroup
563 }

```

$$\iiint_V \iiint_V \iiint_V$$

```

\def\dmath#1{$\displaystyle #1$}
\setmathfont{Cambria Math} \dmath{\iiint_V}
\removenolimits\iiint
\setmathfont{Cambria Math} \dmath{\iiint_V}
\addnolimits\iiint
\setmathfont{Cambria Math} \dmath{\iiint_V}

```

6.5 Radicals

The radical for square root is organised in `\um@set@mathsymbol` on page ?? . I think it's the only radical ever. (Actually, there is also `\cuberoom` and `\fourthroot`, but they don't seem to behave as proper radicals.)

Also, what about right-to-left square roots?

`\um@radicals` We organise radicals in the same way as `nolimits`-operators; that is, in a comma-list.

```

564 \def\um@radicals{\sqrt}

```

$$\sqrt{1 + \sqrt{1 + \sqrt{1 + \sqrt{1 + \sqrt{1 + \sqrt{1 + \sqrt{1 + x}}}}}}}$$

```

\setmathfont{Cambria Math}
\[ \sqrt{1+\sqrt{1+
\sqrt{1+ \sqrt{1+
\sqrt{1+\sqrt{1+
\sqrt{1+x}}}}}} \]

```

$$\sqrt[2]{1 + \sqrt[3]{1 + x}}$$

```

\setmathfont{Cambria Math}
\[ \sqrt[2]{1+\sqrt[3]{1+x}} \]

```

6.6 Delimiters

`\left` We redefine the primitive to be preceded by `\mathopen`; this gives much better spacing in cases such as `\sin\left....` Courtesy of Frank Mittelbach:

<http://www.latex-project.org/cgi-bin/ltbugs2html?pr=latex/3853&prlatex/3754>

```
565 \let\left@primitive\left
566 \def\left{\mathopen{}\left@primitive}
```





















No re-definition is made for `\right` because I don't believe it to be necessary.

: TODO: 'fences', e.g., `\vert`


$\left(\left(\left(\left(x^1\right)^2\right)^3\right)^4\right)^5$	<code>\setmathfont{Cambria Math}</code> <code>\[\left(\left(\left(\left(\left(x</code> <code>\right)^1\right)^2\right)^3\right)^4\right)^5 \]</code>
$\left[\left[\left[\left[\left[y^1\right]^2\right]^3\right]^4\right]^5\right]$	<code>\[\left[\left[\left[\left[\left(y</code> <code>\right]^1\right]^2\right]^3\right]^4\right]^5 \]</code>
$\left\{\left\{\left\{\left\{\left[z^1\right]^2\right]^3\right]^4\right]^5\right\}$	<code>\[\left\{\left\{\left\{\left\{\left(z</code> <code>\right\}^1\right\}^2\right\}^3\right\}^4\right\}^5 \]</code>

Here are all `\mathopen` characters:

USV	Ex.	Macro	Description
U+00028	(<code>\lparen</code>	LEFT PARENTHESIS
U+0005B	[<code>\lbrack</code>	LEFT SQUARE BRACKET
U+0007B	{	<code>\lbrace</code>	LEFT CURLY BRACKET
U+000AB	«	<code>\guillemotleft</code>	(GUILLEMET), LEFT
U+02018	‘	<code>\lq</code>	SINGLE QUOTATION MARK, LEFT
U+0201A	,	<code>\quotsinglbase</code>	RISING SINGLE QUOTE, LEFT (LOW)
U+0201E	„	<code>\quotdblbase</code>	RISING DOUBLE QUOTE, LEFT (LOW)
U+02039	<	<code>\guilsinglleft</code>	(GUILLEMET), LEFT
U+0221A	√	<code>\sqrt</code>	RADICAL
U+0221B	∛	<code>\cuberoot</code>	CUBE ROOT
U+0221C	∜	<code>\fourthroot</code>	FOURTH ROOT
U+02308	⌈	<code>\lceil</code>	LEFT CEILING
U+0230A	⌋	<code>\lfloor</code>	LEFT FLOOR
U+0231C	⌵	<code>\ulcorner</code>	UPPER LEFT CORNER
U+0231E	⌴	<code>\llcorner</code>	LOWER LEFT CORNER
			LIGHT LEFT TORTOISE SHELL BRACKET
U+02772		<code>\lbrbrak</code>	ORNAMENT
U+027C5	⌢	<code>\lbag</code>	LEFT S-SHAPED BAG DELIMITER

U+027CC		<code>\longdivision</code>	LONG DIVISION MATHEMATICAL LEFT WHITE SQUARE
U+027E6		<code>\lBrack</code>	BRACKET
U+027E8		<code>\langle</code>	MATHEMATICAL LEFT ANGLE BRACKET MATHEMATICAL LEFT DOUBLE ANGLE
U+027EA		<code>\lAngle</code>	BRACKET MATHEMATICAL LEFT WHITE TORTOISE
U+027EC		<code>\lbrbrak</code>	SHELL BRACKET
U+02983		<code>\lBrace</code>	LEFT WHITE CURLY BRACKET
U+02985		<code>\lParen</code>	LEFT WHITE PARENTHESIS
U+02987		<code>\llparenthesis</code>	Z NOTATION LEFT IMAGE BRACKET
U+02989		<code>\llangle</code>	Z NOTATION LEFT BINDING BRACKET
U+0298B		<code>\lbrackubar</code>	LEFT SQUARE BRACKET WITH UNDERBAR LEFT SQUARE BRACKET WITH TICK IN TOP
U+0298D		<code>\lbrackultick</code>	CORNER LEFT SQUARE BRACKET WITH TICK IN
U+0298F		<code>\lbracklltick</code>	BOTTOM CORNER
U+02991		<code>\langedot</code>	LEFT ANGLE BRACKET WITH DOT
U+02993		<code>\lparenless</code>	LEFT ARC LESS-THAN BRACKET
U+02997		<code>\lblkbrbrak</code>	LEFT BLACK TORTOISE SHELL BRACKET
U+029D8		<code>\lvzigzag</code>	LEFT WIGGLY FENCE
U+029DA		<code>\Lvzigzag</code>	LEFT DOUBLE WIGGLY FENCE
U+029FC		<code>\lcurvyangle</code>	LEFT POINTING CURVED ANGLE BRACKET
U+03014		<code>\lbrbrak</code>	LEFT BROKEN BRACKET
U+03018		<code>\Lbrbrak</code>	LEFT WHITE TORTOISE SHELL BRACKET

And `\mathclose`:

USV	Ex.	Macro	Description
U+00029)	<code>\rparen</code>	RIGHT PARENTHESIS
U+0005D]	<code>\rbrack</code>	RIGHT SQUARE BRACKET
U+0007D	}	<code>\rbrace</code>	RIGHT CURLY BRACKET DOUBLE ANGLE QUOTATION MARK
U+000BB	»	<code>\guillemotright</code>	(GUILLEMET), RIGHT
U+02019	'	<code>\rq</code>	SINGLE QUOTATION MARK, RIGHT
U+0201B	ˆ	<code>\quotsinglright</code>	RISING SINGLE QUOTE, RIGHT (HIGH)
U+0201F	“	<code>\quotdblright</code>	RISING DOUBLE QUOTE, RIGHT (HIGH) SINGLE ANGLE QUOTATION MARK
U+0203A	>	<code>\guilsinglright</code>	(GUILLEMET), RIGHT
U+02309	⌈	<code>\rceil</code>	RIGHT CEILING
U+0230B	⌋	<code>\rfloor</code>	RIGHT FLOOR
U+0231D	⌵	<code>\urcorner</code>	UPPER RIGHT CORNER
U+0231F	⌴	<code>\lrcorner</code>	LOWER RIGHT CORNER LIGHT RIGHT TORTOISE SHELL BRACKET
U+02773		<code>\rbrbrak</code>	ORNAMENT

U+027C6	⏏	\rbag	RIGHT S-SHAPED BAG DELIMITER MATHEMATICAL RIGHT WHITE SQUARE
U+027E7	⏏	\rBrack	BRACKET
U+027E9	⏏	\rangle	MATHEMATICAL RIGHT ANGLE BRACKET MATHEMATICAL RIGHT DOUBLE ANGLE
U+027EB	⏏	\rAngle	BRACKET MATHEMATICAL RIGHT WHITE TORTOISE
U+027ED		\Rbrbrak	SHELL BRACKET
U+02984	⏏	\rBrace	RIGHT WHITE CURLY BRACKET
U+02986	⏏	\rParen	RIGHT WHITE PARENTHESIS
U+02988	⏏	\rrparenthesis	Z NOTATION RIGHT IMAGE BRACKET
U+0298A	⏏	\rrangle	Z NOTATION RIGHT BINDING BRACKET
U+0298C	⏏	\rbrackubar	RIGHT SQUARE BRACKET WITH UNDERBAR RIGHT SQUARE BRACKET WITH TICK IN
U+0298E	⏏	\rbracklrtick	BOTTOM CORNER RIGHT SQUARE BRACKET WITH TICK IN TOP
U+02990	⏏	\rbrackurtick	CORNER
U+02992	⏏	\rangledot	RIGHT ANGLE BRACKET WITH DOT
U+02994	⏏	\rpangtr	RIGHT ARC GREATER-THAN BRACKET
U+02998	⏏	\rblbrbrak	RIGHT BLACK TORTOISE SHELL BRACKET
U+029D9	⏏	\rvzigzag	RIGHT WIGGLY FENCE
U+029DB	⏏	\Rvzigzag	RIGHT DOUBLE WIGGLY FENCE
U+029FD	⏏	\rcurvyangle	RIGHT POINTING CURVED ANGLE BRACKET
U+03015		\rbrbrak	RIGHT BROKEN BRACKET
U+03019		\Rbrbrak	RIGHT WHITE TORTOISE SHELL BRACKET

6.7 Maths accents

Maths accents should just work *if they are available in the font*.

USV	Ex.	Macro	Description
U+00300	̀	\grave	GRAVE ACCENT
U+00301	́	\acute	ACUTE ACCENT
U+00302	̂	\hat	CIRCUMFLEX ACCENT
U+00303	̃	\tilde	TILDE
U+00304	̄	\bar	MACRON
U+00305	̅	\overbar	OVERBAR EMBELLISHMENT
U+00306	̆	\breve	BREVE
U+00307	̇	\dot	DOT ABOVE
U+00308	̈	\ddot	DIERESIS
U+00309	̉	\ovhook	COMBINING HOOK ABOVE
U+0030A	̊	\ocirc	RING
U+0030C	̌	\check	CARON
U+00310	̎	\candra	CANDRABINDU (NON-SPACING)

U+00312		<code>\turnedcomma</code>	COMBINING TURNED COMMA ABOVE
U+00313		<code>\osmooth</code>	GREEK PSILI (SMOOTH BREATHING) (NON-SPACING)
U+00314		<code>\orough</code>	GREEK DASIA (ROUGH BREATHING) (NON-SPACING)
U+00315		<code>\ocommatopright</code>	COMBINING COMMA ABOVE RIGHT
U+0031A		<code>\droang</code>	LEFT ANGLE ABOVE (NON-SPACING)
U+020D0		<code>\leftharpoonaccent</code>	COMBINING LEFT HARPOON ABOVE
U+020D1		<code>\rightharpoonaccent</code>	COMBINING RIGHT HARPOON ABOVE
U+020D2		<code>\vertoverlay</code>	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6		<code>\overleftarrow</code>	COMBINING LEFT ARROW ABOVE
U+020D7		<code>\vec</code>	COMBINING RIGHT ARROW ABOVE
U+020DB		<code>\dddot</code>	COMBINING THREE DOTS ABOVE
U+020DC		<code>\ddddot</code>	COMBINING FOUR DOTS ABOVE
U+020E1		<code>\overleftrightarrow</code>	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7		<code>\annuity</code>	COMBINING ANNUITY SYMBOL
U+020E8		<code>\threeunderdot</code>	COMBINING TRIPLE UNDERDOT
U+020E9		<code>\widebridgeabove</code>	COMBINING WIDE BRIDGE ABOVE
U+020EC		<code>\underrightharpoondown</code>	BARB DOWNWARDS COMBINING LEFTWARDS HARPOON WITH
U+020ED		<code>\underleftharpoondown</code>	BARB DOWNWARDS
U+020EE		<code>\underleftarrow</code>	COMBINING LEFT ARROW BELOW
U+020EF		<code>\underrightarrow</code>	COMBINING RIGHT ARROW BELOW
U+020F0		<code>\asteraccent</code>	COMBINING ASTERISK ABOVE

7 Font features

`\um@zf@feature` Use the same method as `fontspec` for feature definition (*i.e.*, using `xkeyval`) but with a conditional to restrict the scope of these features to unicode-math commands.

```

567 \newcommand\um@zf@feature[2]{
568   \define@key[zf]{options}{#1}[]{
569     \if@um@fontspec@feature
570       #2
571     \else
572       \PackageError{fontspec/unicode-math}
573         {The ‘#1’ font feature can only be used for maths fonts}
574         {The feature you tried to use can only be in commands
575          like \protect\setmathfont}
576     \fi
577   }
578 }
```

7.1 OpenType maths font features

```
579 \um@zf@feature{ScriptStyle}{  
580   \zf@update@ff{+ssty=0}  
581 }  
582 \um@zf@feature{ScriptScriptStyle}{  
583   \zf@update@ff{+ssty=1}  
584 }
```

7.2 Script and scriptscript font options

```
585 \define@cmdkey[um]{options}[um@]{ScriptFeatures}{}  
586 \define@cmdkey[um]{options}[um@]{ScriptScriptFeatures}{}  
587 \define@cmdkey[um]{options}[um@]{ScriptFont}{}  
588 \define@cmdkey[um]{options}[um@]{ScriptScriptFont}{}  
589
```

7.3 Range processing

The ‘ALL’ branch here is deprecated and happens automatically.

```
589 \define@choicekey+{um}{options}{Range}[\@tempa\@tempb]{ALL}{  
590   \ifcase\@tempb\relax  
591     \global\let\um@char@range\@empty  
592   \fi  
593 }{  
594   \xdef\um@char@range{#1}  
595 }
```

Pretty basic comma separated range processing. Donald Arseneau’s selectp package has a cleverer technique.

`\um@parse@term` #1 : unicode character slot
#2 : control sequence (character macro)
#3 : control sequence (math type)
#4 : code to execute

This macro expands to #4 if any of its arguments are contained in the commalist `\um@char@range`. This list can contain either character ranges (for checking with #1) or control sequences. These latter can either be the command name of a specific character, *or* the math type of one (*e.g.*, `\mathbin`).

Character ranges are passed to `\um@parse@range`, which accepts input in the form shown in table 9.

Start by iterating over the commalist, ignoring empties, and initialising the scratch conditional:

```
596 \newcommand\um@parse@term[4]{  
597   \clist_map_variable:NNn \um@char@range \@ii {  
598     \unless\ifx\@ii\@empty  
599       \@tempswafalse
```

Table 9: Ranges accepted by `\um@parse@range`.

Input	Range
x	$r = x$
$x-$	$r \geq x$
$-y$	$r \leq y$
$x-y$	$x \leq r \leq y$

Match to either the character macro (`\alpha`) or the math type (`\mathbin`):

```

600     \expandafter\um@firstchar\expandafter{\@ii}
601     \ifx\@tempa\um@backslash
602         \expandafter\ifx\@ii#2\relax
603             \@tempswatrue
604         \else
605             \expandafter\ifx\@ii#3\relax
606                 \@tempswatrue
607             \fi
608         \fi

```

Otherwise, we have a number range, which is passed to another macro:

```

609     \else
610         \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
611     \fi

```

If we have a match, execute the code! It also populates the `\um@char@num@range` macro, which is used when defining `\mathbf` (*etc.*) `\mathchar` remappings.

```

612     \if@tempswa
613         \ifx\um@char@num@range\@empty
614             \g@addto@macro\um@char@num@range{#1}
615         \else
616             \g@addto@macro\um@char@num@range{, #1}
617         \fi
618         #4%
619     \fi
620 \fi
621 }
622 }
623 \def\um@firstof#1#2\@nil{#1}
624 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
625 \def\um@firstchar#1{\edef\@tempa{\expandafter\um@firstof\string#1\@nil}}

```

'1' or '\a' or '\b' is included '1' or '\b' or '\c' is included '3' or '\a' or '\b' is included '3' or '\a' or '\b' is included

```
\def\um@char@range{\a,2-4,\c}
\um@parse@term{1}{\a}{\b}
  {\a' or '\string'a' or '\string'b' is included}
\um@parse@term{1}{\b}{\c}
  {\b' or '\string'b' or '\string'c' is included}
\um@parse@term{3}{\a}{\b}
  {\a' or '\string'a' or '\string'b' is included}
```

\um@parse@range Weird syntax. As shown previously in table 9, this macro can be passed four different input types via \um@parse@term.

```
626 \def\um@parse@range#1-#2-#3\@nil#4\@nil{
627   \def\@tempa{#1}
628   \def\@tempb{#2}
Range       $r = x$ 
C-list input \@ii=X
Macro input \um@parse@range X-\@marker-\@nil#1\@nil
Arguments    $\textcolor{red}{\#1}-\textcolor{blue}{\#2}-\textcolor{green}{\#3} = \textcolor{red}{X}-\@marker-\{\}$ 
629   \expandafter\ifx\expandafter\@marker\@tempb\relax
630     \ifnum#4=#1\relax
631       \@tempswatruetrue
632     \fi
633   \else
Range       $r \geq x$ 
C-list input \@ii=X-
Macro input \um@parse@range X--\@marker-\@nil#1\@nil
Arguments    $\textcolor{red}{\#1}-\textcolor{blue}{\#2}-\textcolor{green}{\#3} = \textcolor{red}{X}-\{\}-\@marker-$ 
634     \ifx\@empty\@tempb
635       \ifnum#4>\numexpr#1-1\relax
636         \@tempswatruetrue
637       \fi
638     \else
Range       $r \leq y$ 
C-list input \@ii=-Y
Macro input \um@parse@range -Y-\@marker-\@nil#1\@nil
Arguments    $\textcolor{red}{\#1}-\textcolor{blue}{\#2}-\textcolor{green}{\#3} = \{\}-\textcolor{red}{Y}-\@marker-$ 
639     \ifx\@empty\@tempa
640       \ifnum#4<\numexpr#2+1\relax
641         \@tempswatruetrue
642       \fi
Range       $x \leq r \leq y$ 
C-list input \@ii=X-Y
Macro input \um@parse@range X-Y-\@marker-\@nil#1\@nil
Arguments    $\textcolor{red}{\#1}-\textcolor{blue}{\#2}-\textcolor{green}{\#3} = \textcolor{red}{X}-\textcolor{blue}{Y}-\@marker-$ 
```

```

643     \else
644         \ifnum#4>\numexpr#1-1\relax
645         \ifnum#4<\numexpr#2+1\relax
646             \@tempwattrue
647         \fi
648     \fi
649 \fi
650 \fi
651 \fi
652 }

\um_map_char:nn #1 : Number of iterations
                #2 : Starting input char(s)
                #3 : Starting output char
                Loops through character ranges setting \mathcode.
653 \cs_set:Nn \um_map_chars_range:nnn {
654     \clist_map_variable:nnN {#2} \l_um_input_num {
655         \prg_stepwise_variable:nnnNn{0}{1}{#1} \l_um_incr_num {
656             \um_set_mathcode:nnnn
657             {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
658             {\mathalpha}{\um_symfont_tl}
659             {\numexpr \l_um_incr_num + #3 \relax}
660         }
661     }
662 }
663 \cs_set:Nn \um_map_chars_latin:nn {
664     \um_map_chars_range:nnn {25}{#1}{#2}
665 }
666 \cs_set:Nn \um_map_chars_greek:nn {
667     \um_map_chars_range:nnn {24}{#1}{#2}
668 }
669 \cs_set:Nn \um_map_chars_numbers:nn {
670     \um_map_chars_range:nnn {9}{#1}{#2}
671 }
672 \cs_set:Nn \um_map_char:nn {
673     \um_map_chars_range:nnn {0}{#1}{#2}
674 }

```

```

\um_set_mathalphabet_char:Nnnn #1 : Maths alphabet
                                #2 : Input char(s)
                                #3 : Output char
                                Loops through character ranges setting \mathcode.
675 \cs_set:Npn \exp_args:Nnff {\::n\::f\::f\:::}
676 \cs_new:Nn \um_set_mathalphabet_char:Nnn {
677     \clist_map_variable:nnN {#2} \l_um_input_num {
678         \exp_args:Nnff \um_mathmap:Nnn {#1}

```

```

679     {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
680   }
681 }

```

`\um_set_mathalph_range:Nnn` [*(Number of iterations)*] #1 : Maths alphabet
 #2 : Starting input char(s)
 #3 : Starting output char
 Loops through character ranges setting `\mathcode`.

```

682 \cs_new:Nn \um_set_mathalph_range:nNnn {
683   \clist_map_variable:nNn {#3} \l_um_input_num {
684     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
685       \exp_args:Nnff \um_mathmap:Nnn {#2}
686       {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
687       {\number\numexpr \l_um_inc_num + #4 \relax}
688     }
689   }
690 }
691 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
692   \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
693 }
694 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
695   \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
696 }
697 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
698   \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
699 }

```

BCDBCDEABCDEFG

```

\ExplSyntaxOn
{\um_map_chars_range:nnn{3}{'\A,'\D}{'\B}
$ABCDEF$} $ABCDEF$

```

`\um@resolve@greek` This macro defines `\Alpha...``\omega` as their corresponding unicode (mathematical italic) character. Remember that the mapping to upright or italic happens with the `mathcode` definitions, whereas these macros just stand for the literal unicode characters.

```

700 \AtBeginDocument{\um@resolve@greek}
701 \newcommand\um@resolve@greek{
702   \def\Alpha{\mitAlpha}
703   \def\Beta{\mitBeta}
704   \def\Gamma{\mitGamma}
705   \def\Delta{\mitDelta}
706   \def\Epsilon{\mitEpsilon}
707   \def\Zeta{\mitZeta}
708   \def\Eta{\mitEta}

```



```

709 \def\Theta{\mitTheta}
710 \def\Iota{\mitIota}
711 \def\Kappa{\mitKappa}
712 \def\Lambda{\mitLambda}
713 \def\Mu{\mitMu}
714 \def\Nu{\mitNu}
715 \def\Xi{\mitXi}
716 \def\Omicron{\mitOmicron}
717 \def\Pi{\mitPi}
718 \def\Rho{\mitRho}
719 \def\varTheta{\mitvarTheta}
720 \def\Sigma{\mitSigma}
721 \def\Tau{\mitTau}
722 \def\Upsilon{\mitUpsilon}
723 \def\Phi{\mitPhi}
724 \def\Chi{\mitChi}
725 \def\Psi{\mitPsi}
726 \def\Omega{\mitOmega}

```

Lowercase:

```

727 \def\alpha{\mitalpha}
728 \def\beta{\mitbeta}
729 \def\gamma{\mitgamma}
730 \def\delta{\mitdelta}
731 \def\epsilon{
732   \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
733 }
734 \def\zeta{\mitzeta}
735 \def\eta{\miteta}
736 \def\theta{\mittheta}
737 \def\iota{\mitiota}
738 \def\kappa{\mitkappa}
739 \def\lambda{\mitlambda}
740 \def\mu{\mitmu}
741 \def\nu{\mitnu}
742 \def\xi{\mitxi}
743 \def\omicron{\mitomicron}
744 \def\pi{\mitpi}
745 \def\rho{\mitrho}
746 \def\varsigma{\mitvarsigma}
747 \def\sigma{\mitsigma}
748 \def\tau{\mittau}
749 \def\upsilon{\mitupsilon}
750 \def\phi{
751   \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}
752 }
753 \def\chi{\mitchi}

```

```

754 \def\psi{\mitpsi}
755 \def\omega{\mitomega}
756 \def\varepsilon{
757     \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
758 }
759 \def\vartheta{\mitvartheta}
760 \def\varkappa{\mitvarkappa}
761 \def\varphi{
762     \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
763 }
764 \def\varrho{\mitvarrho}
765 \def\varpi{\mitvarpi}
766 }

```

\um_setup_literals: : TODO : other literal symbols

```

767 \cs_set:Nn \um_setup_literals: {
768     \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}
769     \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}
770     \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}
771     \um_map_char:nn {\um@usv@ith}{\um@usv@ith}
772     \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}
773     \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}
774     \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}
775     \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}
776     \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}
777 }

```

\um_setup_bf_literals: TODO: other literal symbols

```

778 \cs_set:Nn \um_setup_bf_literals: {
779     \um_map_chars_latin:nn {\um@usv@bfupLatin}{\um@usv@bfupLatin}
780     \um_map_chars_latin:nn {\um@usv@bfuplatin}{\um@usv@bfuplatin}
781     \um_map_chars_latin:nn {\um@usv@bfitLatin}{\um@usv@bfitLatin}
782     \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}
783     \um_map_chars_greek:nn {\um@usv@bfupGreek}{\um@usv@bfupGreek}
784     \um_map_chars_greek:nn {\um@usv@bfupgreek}{\um@usv@bfupgreek}
785     \um_map_chars_greek:nn {\um@usv@bfitGreek}{\um@usv@bfitGreek}
786     \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
787 }

```

\um_setup_Latin:

```

788 \cs_set:Nn \um_setup_Latin: {
789     \if@um@upLatin
790         \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
791     \else
792         \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
793     \fi
794 }

```

\um_setup_latin: Don't overlook 'h', which maps to U+210E: PLANCK CONSTANT instead of the expected U+1D455: MATHEMATICAL ITALIC SMALL H.

```

795 \cs_set:Nn \um_setup_latin: {
796   \if@um@uplatin
797     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
798     \um_map_char:nn {\um@usv@ith}{`\h}
799   \else
800     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
801     \um_map_char:nn {\h,\um@usv@ith}{\um@usv@ith}
802   \fi
803 }

```

\um_setup_Greek:

```

804 \cs_set:Nn \um_setup_Greek: {
805   \if@um@upGreek
806     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
807     \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
808   \else
809     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
810     \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}
811   \fi
812 }

```

\um_setup_greek:

```

813 \cs_set:Nn \um_setup_greek: {
814   \if@um@upgreek
815     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
816     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
817     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
818     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
819     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
820     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
821     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
822   \else
823     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
824     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
825     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
826     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
827     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
828     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
829     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
830   \fi
831 }

```

8 Maths alphabets mapping definitions

Algorithm for setting alphabet fonts:

- By default, try and set all of them.
- Check for the first glyph of each to detect if the font supports each alphabet. (This doesn't work to distinguish Latin/Greek but we hope all maths fonts will have at least them!)
- For alphabets that are not supported, *do nothing*. (This includes leaving the old alphabet definition in place.)
- For alphabets that do exist, overwrite whatever's already there.

```

832 \cs_new:Nn \um_setup_math_alphabet:n {
833   \um_glyph_if_exist:nTF {\csname um@usv@#1latin \endcsname}{
834     \um_maybe_init_alphabet:n {#1}
835     \um_prepare_alph:n {#1}
836     \use:c {um_config_math#1:}
837   }{
838     \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
839       Math~ alphabet~ \@backslashchar math#1~ not~ found~ in~ font~ \font-
840       name\um@font}
841     \cs_if_exist:cT {um_fix_math#1:} {
842       \use:c {um_fix_math#1:}
843     }
844   }
845 \cs_set:Nn \um_fix_mathtt: {
846   \SetMathAlphabet\mathtt{normal}\encodingdefault\ttdefault\mddefault\updefault
847 }
848 \cs_set:Nn \um_init_alphabet:n {
849   \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
850 }

```

`\um_glyph_if_exist:nTF` : TODO: Generalise for arbitrary fonts! `\um@font` is not always the one used for a specific glyph!!

```

851 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
852   \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
853 }

```

`\um_prepare_alph:n` If `\mathXY` hasn't been (re-)declared yet, then define it in terms of `unicode-math` definitions. Use `\bgroup/\egroup` so s'scripts scan the whole thing.

```

854 \cs_new:Nn \um_prepare_alph:n {
855   \cs_if_exist:cF {um_math#1:n} {
856     \cs_set:cpn {um_math#1:n} ##1 {

```

```

857     \use:c {um_setup_math#1:} ##1 \egroup
858   }
859   \cs_set_protected:cpn {math#1} {
860     \bgroup
861     \mode_if_math:F {
862       \egroup\expandafter
863       \non@alpherr\expandafter{\csname math#1\endcsname\space}
864     }
865     \use:c {um_math#1:n}
866   }
867 }
868 }

869 \cs_new:Nn \um_setup_alphabets: {
870   \um_setup_math_alphabet:n {up   }
871   \um_setup_math_alphabet:n {it   }
872   \um_setup_math_alphabet:n {bb   }
873   \um_setup_math_alphabet:n {scr  }
874   \um_setup_math_alphabet:n {frak }
875   \um_setup_math_alphabet:n {sf   }
876   \um_setup_math_alphabet:n {sfup }
877   \um_setup_math_alphabet:n {sfit }
878   \um_setup_math_alphabet:n {tt   }
879   \um_setup_math_alphabet:n {bf   }
880   \um_setup_math_alphabet:n {bfup }
881   \um_setup_math_alphabet:n {bfit }
882   \um_setup_math_alphabet:n {bfscr }
883   \um_setup_math_alphabet:n {bffrak}
884   \um_setup_math_alphabet:n {bfsf }
885   \um_setup_math_alphabet:n {bfsfup}
886   \um_setup_math_alphabet:n {bfsfit}
887 }

```

: TODO : nested alphabets?

8.0.1 Upright: `\mathup`

ABCDEFGHJKLMNOPQRSTUVWXYZ	<code>\mathup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \</code>
abcdefghijklmnopqrstuvwxyz	<code>\mathup{abcdefghijklmnopqrstuvwxyz}\$ \</code>
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	<code>\mathup{ }\$ \quad\mathup{ }\$ \</code>
αβγδεζηθικλμνξοπρστυφχψω εθκφρϖ	<code>\mathup{ }\$ \quad\mathup{ }\$ \</code>

Takes both upright and italic characters to be typeset as upright symbols.

```

888 \cs_new:Npn \um_config_mathup: {
889   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}

```

```

890 \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
891 \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
892 \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
893 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}
894 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}
895 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varTheta}
896 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
897 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
898 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
899 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
900 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
901 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
902 }

```

8.0.2 Italic: \mathit

<i>ABCDEFGHIJKLMNOPQRSTUVWXYZ</i>	$\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
<i>abcdefghijklmnopqrstuvwxyz</i>	$\mathit{abcdefghijklmnopqrstuvwxyz}$
<i>ΑΒΓΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ</i>	$\mathit{ΑΒΓΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ}$
<i>αβγδεζηθικλμνξοπρστυφχψω εθικρρ</i>	$\mathit{αβγδεζηθικλμνξοπρστυφχψω εθικρρ}$

Roman:

```

903 \cs_new:Npn \um_config_mathit: {
904 \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
905 \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
906 \um_set_mathalphabet_char:Nnn{\mathit}{\h,\um@usv@ith}{\um@usv@ith}

```

Greek:

```

907 \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
908 \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
909 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
910 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartial}
911 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarTheta}
912 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
913 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
914 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
915 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
916 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
917 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
918 }

```

8.0.3 Blackboard or double-struck: `\mathbb`

	0123456789	<code>\$\mathbb{0123456789}\$ \\\</code>
ABCDEFGHIJKLMNOPQRSTUVWXYZ		<code>\$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\\</code>
abcdefghijklmnopqrstuvwxyz		<code>\$\mathbb{abcdefghijklmnopqrstuvwxyz}\$ \\\</code>

Numbers:

```
919 \cs_new:Npn \um_config_mathbb: {
920   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@enum}{\um@usv@bnum}
```

Roman uppercase:

```
921 \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bblatin}
922 \um_set_mathalphabet_char:Nnn{\mathbb}{`\C,"1D60A}{`"2102}
923 \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D60F}{`"210D}
924 \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D60F}{`"2115}
925 \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D617}{`"2119}
926 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D618}{`"211A}
927 \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D619}{`"211D}
928 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D621}{`"2124}
```

Roman lowercase:

```
929 \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}
930 }
```

8.0.4 Script or caligraphic: `\mathscr` and `\mathcal`

<i>ABCDEFGHIJKLMNOPQRSTUVWXYZ</i>	<code>\$\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\\</code>
<i>abcdefghijklmnopqrstuvwxyz</i>	<code>\$\mathscr{abcdefghijklmnopqrstuvwxyz}\$ \\\</code>

```
931 \cs_new:Npn \um_config_mathscr: {
932   \um_set_mathalphabet_latin:Nnn{\mathscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}
933   \um_set_mathalphabet_char:Nnn{\mathscr}{`\B,"1D435}{`"212C}
934   \um_set_mathalphabet_char:Nnn{\mathscr}{`\E,"1D438}{`"2130}
935   \um_set_mathalphabet_char:Nnn{\mathscr}{`\F,"1D439}{`"2131}
936   \um_set_mathalphabet_char:Nnn{\mathscr}{`\H,"1D43B}{`"210B}
937   \um_set_mathalphabet_char:Nnn{\mathscr}{`\I,"1D43C}{`"2110}
938   \um_set_mathalphabet_char:Nnn{\mathscr}{`\L,"1D43F}{`"2112}
939   \um_set_mathalphabet_char:Nnn{\mathscr}{`\M,"1D440}{`"2133}
940   \um_set_mathalphabet_char:Nnn{\mathscr}{`\R,"1D445}{`"211B}
941   \um_set_mathalphabet_latin:Nnn{\mathscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrLatin}
942   \um_set_mathalphabet_char:Nnn{\mathscr}{`\e,"1D452}{`"212F}
943   \um_set_mathalphabet_char:Nnn{\mathscr}{`\g,"1D454}{`"210A}
944   \um_set_mathalphabet_char:Nnn{\mathscr}{`\o,"1D45C}{`"2134}
945 }
```

8.0.5 Fraktur or fraktur or blackletter: \mathfrak

$\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
 $\mathfrak{abcdefghijklmnopqrstuvwxyz}$

$\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
 $\mathfrak{abcdefghijklmnopqrstuvwxyz}$ \\

Letters, with exceptions $\{\mathfrak{C}, \mathfrak{S}, \mathfrak{Z}, \mathfrak{H}, \mathfrak{J}\}$:

```

946 \cs_new:Npn \um_config_mathfrak: {
947   \um_set_mathalphabet_latin:Nnn{\mathfrak}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLatin}
948   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\C,"1D436}{`"212D}
949   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\H,"1D43B}{`"210C}
950   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\I,"1D43C}{`"2111}
951   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\R,"1D445}{`"211C}
952   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\Z,"1D44D}{`"2128}
953   \um_set_mathalphabet_latin:Nnn{\mathfrak}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklatin}
954 }

```

8.0.6 Sans serif: \mathsf

$\mathsf{0123456789}$
 $\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
 $\mathsf{abcdefghijklmnopqrstuvwxyz}$

$\mathsf{0123456789}$ \\
 $\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
 $\mathsf{abcdefghijklmnopqrstuvwxyz}$ \\

```

955 \cs_new:Npn \um_config_mathsf: {
956   \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@enum}{\um@usv@sfnun}
957   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfulatin}
958   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfulatin}
959 }

```

8.0.7 Sans serif italic: \mathsf{it}

$\mathsf{0123456789}$
 $\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
 $\mathsf{abcdefghijklmnopqrstuvwxyz}$

$\mathsf{0123456789}$ \\
 $\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
 $\mathsf{abcdefghijklmnopqrstuvwxyz}$ \\

```

960 \cs_new:Npn \um_config_mathsf{it}: {
961   \um_set_mathalphabet_numbers:Nnn{\mathsf{it}}{\um@usv@enum}{\um@usv@sfnun}
962   \um_set_mathalphabet_latin:Nnn{\mathsf{it}}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLatin}
963   \um_set_mathalphabet_latin:Nnn{\mathsf{it}}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlatin}
964 }

```


8.0.8 Typewriter or monospaced: `\mathtt`

0123456789	$\mathtt{0123456789}$
ABCDEFGHIJKLMN	$\mathtt{ABCDEFGHIJKLMN}$
OPQRSTUVWXYZ	$\mathtt{OPQRSTUVWXYZ}$
abcdefghijklmnopqrstuvwxyz	$\mathtt{abcdefghijklmnopqrstuvwxyz}$

```

965 \cs_new:Npn \um_config_mathtt: {
966   \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@enum}{\um@usv@ettnum}
967   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ettLatin}
968   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ettlatin}
969 }

```

8.1 Bold alphabets' character mappings

8.1.1 Bold: `\mathbf`

0123456789	<code>\$\mathbf{0123456789}\$ \\\</code>
ABCDEFGHIJKLMNOPQRSTUVWXYZ	<code>\$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\\</code>
abcdefghijklmnopqrstuvwxyz	<code>\$\mathbf{abcdefghijklmnopqrstuvwxyz}\$ \\\</code>
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ	<code>\$\mathbf{\text{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}}\$ \\\</code>
Θ	<code>\$\mathbf{\text{Θ}}\$ \quad \$\mathbf{\text{Θ}}\$ \\\</code>
αβγδεζηθικλμνξοπρστυφχψω	<code>\$\mathbf{\text{αβγδεζηθικλμνξοπρστυφχψω}}\$ \quad \$\mathbf{\text{αβγδεζηθικλμνξοπρστυφχψω}}\$ \\\</code>
εθκφο	

```

970 \cs_new:Npn \um_config_mathbf: {
971   \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnun}
972   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bDigamma}
973   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bdigamma}
974   \ifum@bfliteral
975     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bupLatin}
976     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@iitLatin}{\um@usv@bfiitLatin}
977     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@buplatin}
978     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@iitlatin}{\um@usv@bfiitlatin}
979     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bupGreek}
980     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@iitGreek}{\um@usv@bfiitGreek}
981     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bupgreek}
982     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@iitgreek}{\um@usv@bfiitgreek}
983     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfiith}
984     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bvarTheta}
985     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bNabla}
986     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bDigamma}
987     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bpartial}
988     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bvarepsilon}

```

```

989 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
990 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
991 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
992 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
993 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}
994 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
995 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
996 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
997 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
998 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
999 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
1000 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1001 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
1002 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
1003 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
1004 \else
1005 \ifum@bfupLatin
1006 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLatin}
1007 \else
1008 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1009 \fi
1010 \ifum@bfuplatin
1011 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplatin}
1012 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
1013 \else
1014 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
1015 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1016 \fi
1017 \ifum@bfupGreek
1018 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGreek}
1019 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarT}
1020 \else
1021 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
1022 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitva}
1023 \fi
1024 \ifum@bfupgreek
1025 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgreek}
1026 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf}
1027 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvarT}
1028 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvark}
1029 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1030 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1031 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1032 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartia}
1033 \else
1034 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}

```

```

1035 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf}
1036 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitva}
1037 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitva}
1038 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi}
1039 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho}
1040 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1041 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpart}
1042 \fi
1043 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um@bfNabla_up_or_it_}
1044 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@bfpartial_up_or_it_}
1045 \fi
1046 }

```

8.1.2 Bold Italic: \mathbfit

0123456789	$\mathbfit{0123456789}$
ABCDEFGHIJKLMNOPQRSTUVWXYZ	$\mathbfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
abcdefghijklmnopqrstuvwxyz	$\mathbfit{abcdefghijklmnopqrstuvwxyz}$
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	$\mathbfit{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ}$
αβγδεζηθικλμνξοπρστυφχψω εθκφρϖ	$\mathbfit{αβγδεζηθικλμνξοπρστυφχψω εθκφρϖ}$

```

1047 \cs_new:Npn \um_config_mathbfit: {
1048   \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnum}
1049   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1050   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
1051   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
1052   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}
1053   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfupLatin}{\um@usv@bfitLatin}
1054   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfuplatin}{\um@usv@bfitlatin}
1055   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfupGreek}{\um@usv@bfitGreek}
1056   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfupgreek}{\um@usv@bfitgreek}
1057   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitvar}
1058   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}
1059   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpart}
1060   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf}
1061   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitvar}
1062   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitvar}
1063   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi}
1064   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho}
1065   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1066 }

```

8.1.3 Bold Italic: `\mathbfup`

0123456789	<code>\$\mathbfup{0123456789}\$ \\</code>
ABCDEFGHIJKLMNOPQRSTUVWXYZ	<code>\$\mathbfup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\</code>
abcdefghijklmnopqrstuvwxyz	<code>\$\mathbfup{abcdefghijklmnopqrstuvwxyz}\$ \\</code>
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	<code>\$\mathbfup{ }\$ \\</code>
αβγδεζηθικλμνξοπρστυφχψω εθκφρψ	<code>\$\mathbfup{ }\$ \\</code>

```

1067 \cs_new:Npn \um_config_mathbfup: {
1068   \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@enum}{\um@usv@bfnun}
1069   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLatin}
1070   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplatin}
1071   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGreek}
1072   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgreek}
1073   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bfupLatin}{\um@usv@bfupLatin}
1074   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bfuplatin}{\um@usv@bfuplatin}
1075   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfupGreek}{\um@usv@bfupGreek}
1076   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfupgreek}{\um@usv@bfupgreek}
1077   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarTheta}
1078   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}
1079   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartial}
1080   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
1081   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1082   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1083   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1084   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1085   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1086 }

```

8.1.4 Bold fractur or fraktur or blackletter: `\mathbffrak`

ABCDEFGHIJKLMNOPQRSTUVWXYZ	<code>\$\mathbffrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\</code>
abcdefghijklmnopqrstuvwxyz	<code>\$\mathbffrak{abcdefghijklmnopqrstuvwxyz}\$ \\</code>

```

1087 \cs_new:Npn \um_config_mathbffrak: {
1088   \um_set_mathalphabet_numbers:Nnn{\mathbffrak}{\um@usv@enum}{\um@usv@bfnun}
1089   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin, \um@usv@itLatin,\um@usv@frakLatin}
1090   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin,\um@usv@fraklatin}
1091 }

```

8.1.5 Bold script or calligraphic: `\mathbfscr`

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

```
$\mathbfscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfscr{abcdefghijklmnopqrstuvwxyz}$ \\
```

```
1092 \cs_new:Npn \um_config_mathbfscr: {
1093   \um_set_mathalphabet_numbers:Nnn{\mathbfscr}{\um@usv@num}{\um@usv@bfnum}
1094   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrLat:
1095   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrLat:
1096 }
```

8.1.6 Bold sans serif: `\mathbfsf`

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ
αβγδεζηθικλμνξοπρστυφχψω εδκφρω

```
\setmathfont{STIXGeneral-Bold}
$\mathbfsf{0123456789}$ \\
$\mathbfsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfsf{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbfsf{ }$ \\
$\mathbfsf{ }$ \\
$\mathbfsf{ }$ \\
$\mathbfsf{ }$ \\
```

: TODO : These should be contextual!
Numbers (always upright) and letters:

```
1097 \cs_new:Npn \um_config_mathbfsf: {
1098   \um_set_mathalphabet_numbers:Nnn{\mathbfsf}{\um@usv@num}{\um@usv@bfnum}
1099   \um_set_mathalphabet_latin:Nnn{\mathbfsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfupLat:
1100   \um_set_mathalphabet_latin:Nnn{\mathbfsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfuplat:
1101   \um_set_mathalphabet_greek:Nnn{\mathbfsf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfupGree
1102   \um_set_mathalphabet_greek:Nnn{\mathbfsf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfupgree
```

Others:

```
1103 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varTheta,\um@usv@itvarTheta}{ "1D767}
1104 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@Nabla,\um@usv@itNabla}{ "1D76F}
1105 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@partial,\um@usv@itpartial}{ "1D789}
1106 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{ "1D78A}
1107 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@vartheta,\um@usv@itvartheta}{ "1D78B}
1108 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varkappa,\um@usv@itvarkappa}{ "1D78C}
1109 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varphi,\um@usv@itvarphi}{ "1D78D}
1110 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varrho,\um@usv@itvarrho}{ "1D78E}
1111 \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varpi,\um@usv@itvarpi}{ "1D78F}
1112 }
```

8.1.7 Bold upright sans serif: `\mathbfsup`

<p>0123456789</p> <p>ABCDEFGHIJKLMNOPQRSTUVWXYZ</p> <p>abcdefghijklmnopqrstuvwxyz</p> <p>ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ</p> <p>αβγδεζηθικλμνξοπρστυφχψω εδκφρω</p>	<pre>\setmathfont{STIXGeneral-Bold} \$\mathbfsup{0123456789}\$ \ \$\mathbfsup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \ \$\mathbfsup{abcdefghijklmnopqrstuvwxyz}\$ \ \$\mathbfsup{ }\$\quad \$\mathbfsup{ }\$ \ \$\mathbfsup{ }\$\quad \$\mathbfsup{ }\$ \</pre>
---	--

Numbers (always upright) and letters:

```
1113 \cs_new:Npn \um_config_mathbfsup: {
1114   \um_set_mathalphabet_numbers:Nnn{\mathbfsup}{\um@usv@num}{\um@usv@bfnun}
1115   \um_set_mathalphabet_latin:Nnn{\mathbfsup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsupLa
1116   \um_set_mathalphabet_latin:Nnn{\mathbfsup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsupLa
1117   \um_set_mathalphabet_greek:Nnn{\mathbfsup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsupGr
1118   \um_set_mathalphabet_greek:Nnn{\mathbfsup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsupgr}
```

Others:

```
1119   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varTheta,\um@usv@itvarTheta}{1D767}
1120   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@Nabla,\um@usv@itNabla}{1D76F}
1121   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@partial,\um@usv@itpartial}{1D789}
1122   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D78A}
1123   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1124   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1125   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1126   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varrho,\um@usv@itvarrho}{1D78E}
1127   \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1128 }
```

8.1.8 Bold italic sans serif: `\mathbfsfit`

<p><i>ABCDEFGHIJKLMNOPQRSTUVWXYZ</i></p> <p><i>abcdefghijklmnopqrstuvwxyz</i></p> <p><i>ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ</i></p> <p><i>αβγδεζηθικλμνξοπρστυφχψω εδκφρω</i></p>	<pre>\setmathfont{STIXGeneral-BoldItalic} \$\mathbfsfit{0123456789}\$ \ \$\mathbfsfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \ \$\mathbfsfit{abcdefghijklmnopqrstuvwxyz}\$ \ \$\mathbfsfit{ }\$\quad \$\mathbfsfit{ }\$ \ \$\mathbfsfit{ }\$\quad \$\mathbfsfit{ }\$ \</pre>
--	---

```
1129 \cs_new:Npn \um_config_mathbfsfit: {
1130   \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@num}{\um@usv@bfnun}
1131   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfitLa
1132   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfitLa
1133   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfitGr
1134   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfitgr}
```

Other symbols:

```

1135 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{1D7A1}
1136 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNabla}
1137 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsfitpartial}
1138 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7C4}
1139 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{1D7C5}
1140 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{1D7C6}
1141 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{1D7C7}
1142 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{1D7C8}
1143 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{1D7C9}
1144 }

```

8.2 Definitions of the math symbols

Here we define every unicode math codepoint an equivalent macro name. The two are equivalent, in a `\let\xyz=^^^1234` kind of way.

`\um@scancharlet` We need to do some trickery to transform the `\UnicodeMathSymbol` argument `\um@scanactivedef` "ABCDEF into the X_YTeX ‘caret input’ form `^^^abcdef`. It is *very important* that the argument has five characters. Otherwise we need to change the number of ^ chars.

To do this, turn ^ into a regular ‘other’ character and define the macro to perform the lowercasing and `\let`. `\scantokens` changes the carets back into their original meaning after the group has ended and ^’s catcode returns to normal.

```

1145 \begingroup
1146 \char_make_other:N \^
1147 \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1148   \lowercase{
1149     \scantokens{\global\let#1=^^^#2}
1150   }
1151 }

```

Making ^ the right catcode isn’t strictly necessary right now but it helps to future proof us with, e.g., `breqn`.

```

1152 \gdef\um@scanactivedef"#1\@nil#2{
1153   \lowercase{
1154     \tl_rescan:nn{
1155       \char_make_math_superscript:N\^
1156     }{
1157       \global\def^^^#1{#2}
1158     }
1159   }
1160 }
1161 \endgroup

```

Now give `\UnicodeMathSymbol` a definition in terms of `\um@scancharlet` and we're good to go.

```

1162 \begingroup
1163   \def\UnicodeMathSymbol#1#2#3#4{
1164     \um@scancharlet#2=#1\@nil
1165   }
1166   \@input{unicode-math-table.tex}
1167 \endgroup

```

9 Epilogue

Lots of little things to tidy up.

9.0.1 Primes

$$\begin{array}{l} [x'] [x'''] [x'''''] \\ [x'] [x'''] [x'''''] \\ [x'] [x'''] [x'''''] \end{array}$$

```

\setmathfont{Cambria Math}
[$x\prime$] [$x\prime\prime\prime$]
[$x\prime\prime\prime$] [$x\prime\prime\prime\prime$] \~
[$x'$] [$x'''$] [$x''''$] \~
[$x$] [$x'$] [$x''$] [$x'''$]

```

We need a new ‘prime’ algorithm. Unicode math has four pre-drawn prime glyphs.

```

U+2032: PRIME (\primesingle): x'
U+2033: DOUBLE PRIME (\primedouble): x''
U+2034: TRIPLE PRIME (\primetripel): x'''
U+2057: QUADRUPLE PRIME (\primequadruple): x''''

```

As you can see, they're all drawn at the correct height without being superscripted. However, in a correctly behaviour OpenType font with the MATH table, we also see different behaviour after the ssty feature is applied:

```

U+2032: PRIME in the 'scriptstyle' font: x'

```

The shrinking and offsetting is done as it is turned into a superscript. This means, luckily, that by default things work nicely for single primes. We can write `x\prime` or `x^\prime` and get: `x'` and `x'`. To support single primes, then, things are easier than in \LaTeX ; we can just map `'` to `\prime` and not worry about it.

However, it would be nice to use the pre-composed primes above if they exist in the font; consider `x'''` vs. `x'''`. Our algorithm is

- Prime encountered; pcount=1.
- Scan ahead; if prime: pcount:=pcount+1; repeat.

- If not prime, stop scanning.
- If pcount=1, \prime, end.
- If pcount=2, check \primedouble; if it exists, use it, end; if not, goto last step.
- Ditto pcount=3 & \primetriples.
- Ditto pcount=4 & \primequadruple.
- If pcount>4 or the glyph doesn't exist, insert pcount \primes with \primekern between each.

```

1168 \muskip_new:N \g_um_primekern_muskip
1169 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1170 \num_new:N \l_um_primecount_num

1171 \cs_new:Nn \um_nprimes:n {
1172   \primesingle
1173   \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1174 }
1175 \cs_new:Nn \um_nprimes_select:n {
1176   \prg_case_int:nnn {#1}{
1177     {1} { \primesingle }
1178     {2} {
1179       \um_glyph_if_exist:nTF {"2033} {\primedouble} {\um_nprimes:n {#1}}
1180     }
1181     {3} {
1182       \um_glyph_if_exist:nTF {"2034} {\primetriples} {\um_nprimes:n {#1}}
1183     }
1184     {4} {
1185       \um_glyph_if_exist:nTF {"2057} {\primequadruple} {\um_nprimes:n {#1}}
1186     }
1187   }{
1188     \um_nprimes:n {#1}
1189   }
1190 }

```

Scanning is more annoying than you'd think because we want to support all three of \prime, ', and the unicode prime. And \ifx doesn't work with mathactive chars.

Insert a \bgroup...\egroup wrapper so that superscript primes work, but does this break spacing for the rest of the time?

```

1191 \cs_new:Nn \um_scanprime: {
1192   \bgroup
1193   \num_zero:N \l_um_primecount_num
1194   \um_scanprime_collect:
1195 }
1196 \cs_new:Nn \um_scanprime_collect: {

```

```

1197 \num_incr:N \l_um_primecount_num
1198 \peek_meaning_remove:NTF ' {
1199   \um_scanprime_collect:
1200 }{
1201   \peek_meaning_remove:NTF \um_scanprime: {
1202     \um_scanprime_collect:
1203   }{
1204     \peek_meaning_remove:NTF ^^^^2032 {
1205       \um_scanprime_collect:
1206     }{
1207       \um_nprimes_select:n {\l_um_primecount_num}
1208     }
1209   }
1210 }
1211 }
1212 }

1213 \cs_set_eq:NN \prime \um_scanprime:
1214 \group_begin:
1215   \char_make_active:N \prime
1216   \char_make_active:n {"2032}
1217   \cs_gset_eq:NN \prime \um_scanprime:
1218   \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1219 \group_end:

```

9.0.2 Unicode radicals

Undo the damage made to `\sqrt`:

```

1220 \DeclareRobustCommand\sqrt{\@ifnextchar[\@sqrt\sqrtsign}

```

`\r@@t #1` : A `mathstyle` (for `\mathpalette`)

`#2` : Leading superscript for the `\sqrt` sign

A re-implementation of L^AT_EX's hard-coded n-root sign using the appropriate `\fontdimens`.

```

1221 \def\r@@t#1#2{
1222   \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
1223   \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1224   \raise \dimexpr(
1225     \um@fontdimen@percent{65}{\um@font}\ht\z@-
1226     \um@fontdimen@percent{65}{\um@font}\dp\z@
1227   )\relax
1228   \copy \rootbox
1229   \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1230   \box \z@
1231 }

```

9.0.3 Unicode sub- and super-scripts

The idea here is to enter a scanning state after a superscript or subscript is encountered. If subsequent superscripts or subscripts (resp.) are found, they are lumped together. Each sub/super has a corresponding regular size glyph which is used by \XeTeX to typeset the results; this means that the actual subscript/superscript glyphs are never seen in the output document — they are only used as input characters.

Open question: should the superscript-like ‘modifiers’ (U+1D2C: MODIFIER CAPITAL LETTER A and on) be included here?

First, the setup of each mathactive char:

```
1232 \prop_new:N \g_um_supers_prop
1233 \prop_new:N \g_um_subs_prop
1234 \cs_generate_variant:Nn \prop_gput:Nnn {Nxn}
1235 \cs_generate_variant:Nn \prop_get:NnN {cxN}
1236 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}
1237
1238 \group_begin:
1239
1240 % Populate a property list with superscript characters; their mean-
1241 % ing as their key,
1242 % for reasons that will become apparent soon, and their replace-
1243 % ment as each key's value.
1244 % Then make the superscript active and bind it to the scanning function.
1245 %
1246 % \cs{scantokens} makes this process much simpler since we can acti-
1247 % vate the char
1248 % and assign its meaning in one step.
1249 \cs_set:Nn \um_setup_active_superscript:nn {
1250   \prop_gput:Nxn \g_um_supers_prop {\meaning #1} {#2}
1251   \char_make_active:n {'#1}
1252   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1253   \scantokens{
1254     \cs_gset:Npn #1 {
1255       \tl_set:Nn \l_um_ss_chain_tl {#2}
1256       \cs_set_eq:NN \um_sub_or_super:n \sp
1257       \tl_set:Nn \l_um_tmpa_tl {supers}
1258       \um_scan_sscript:
1259     }
1260   }
1261 }
1262
1263 \um_setup_active_superscript:nn {^^^2070} {0}
1264 \um_setup_active_superscript:nn {^^^00b9} {1}
1265 \um_setup_active_superscript:nn {^^^00b2} {2}
1266 \um_setup_active_superscript:nn {^^^00b3} {3}
```

```

1264 \um_setup_active_superscript:nn {^^^2074} {4}
1265 \um_setup_active_superscript:nn {^^^2075} {5}
1266 \um_setup_active_superscript:nn {^^^2076} {6}
1267 \um_setup_active_superscript:nn {^^^2077} {7}
1268 \um_setup_active_superscript:nn {^^^2078} {8}
1269 \um_setup_active_superscript:nn {^^^2079} {9}
1270 \um_setup_active_superscript:nn {^^^207a} {+}
1271 \um_setup_active_superscript:nn {^^^207b} {-}
1272 \um_setup_active_superscript:nn {^^^207c} {=}
1273 \um_setup_active_superscript:nn {^^^207d} {(}
1274 \um_setup_active_superscript:nn {^^^207e} {)}
1275 \um_setup_active_superscript:nn {^^^207i} {i}
1276 \um_setup_active_superscript:nn {^^^207f} {n}
1277
1278 % Ditto above.
1279 \cs_set:Nn \um_setup_active_subscript:nn {
1280   \prop_gput:Nxn \g_um_subs_prop {\meaning #1} {#2}
1281   \char_make_active:n {'#1}
1282   \global\XeTeXmathcodenum '#1 = "1FFFFF \scan_stop:
1283   \scantokens{
1284     \cs_gset:Npn #1 {
1285       \tl_set:Nn \l_um_ss_chain_tl {#2}
1286       \cs_set_eq:NN \um_sub_or_super:n \sb
1287       \tl_set:Nn \l_um_tmpa_tl {subs}
1288       \um_scan_sscript:
1289     }
1290   }
1291 }
1292
1293 \um_setup_active_subscript:nn {^^^2080} {0}
1294 \um_setup_active_subscript:nn {^^^2081} {1}
1295 \um_setup_active_subscript:nn {^^^2082} {2}
1296 \um_setup_active_subscript:nn {^^^2083} {3}
1297 \um_setup_active_subscript:nn {^^^2084} {4}
1298 \um_setup_active_subscript:nn {^^^2085} {5}
1299 \um_setup_active_subscript:nn {^^^2086} {6}
1300 \um_setup_active_subscript:nn {^^^2087} {7}
1301 \um_setup_active_subscript:nn {^^^2088} {8}
1302 \um_setup_active_subscript:nn {^^^2089} {9}
1303 \um_setup_active_subscript:nn {^^^208a} {+}
1304 \um_setup_active_subscript:nn {^^^208b} {-}
1305 \um_setup_active_subscript:nn {^^^208c} {=}
1306 \um_setup_active_subscript:nn {^^^208d} {(}
1307 \um_setup_active_subscript:nn {^^^208e} {)}
1308 \um_setup_active_subscript:nn {^^^2090} {a}
1309 \um_setup_active_subscript:nn {^^^2091} {e}

```

```

1310 \um_setup_active_subscript:nn {^^^1d62} {i}
1311 \um_setup_active_subscript:nn {^^^2092} {o}
1312 \um_setup_active_subscript:nn {^^^1d63} {r}
1313 \um_setup_active_subscript:nn {^^^1d64} {u}
1314 \um_setup_active_subscript:nn {^^^1d65} {v}
1315 \um_setup_active_subscript:nn {^^^2093} {x}
1316 \um_setup_active_subscript:nn {^^^1d66} {\beta}
1317 \um_setup_active_subscript:nn {^^^1d67} {\gamma}
1318 \um_setup_active_subscript:nn {^^^1d68} {\rho}
1319 \um_setup_active_subscript:nn {^^^1d69} {\phi}
1320 \um_setup_active_subscript:nn {^^^1d6a} {\chi}
1321
1322 \group_end:
1323
1324 % The scanning command, evident in its purpose:
1325 \cs_new:Nn \um_scan_sscript: {
1326   \um_scan_sscript:TF {
1327     \um_scan_sscript:
1328   }{
1329     \um_sub_or_super:n {\l_um_ss_chain_tl}
1330   }
1331 }
1332
1333 % The main theme here is stolen from the source to the various \cs{peek_} func-
1334 % tions.
1335 % Consider this function as simply boilerplate:
1336 \cs_new:Nn \um_scan_sscript:TF {
1337   \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
1338   \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
1339   \tl_set:Nx \l_peek_false_tl { \exp_not:n{\group_align_safe_end: #2}}
1340   \group_align_safe_begin:
1341   \peek_after:NN \um_peek_execute_branches_ss:
1342 }
1343
1344 % We do not skip spaces when scanning ahead, and we explicitly wish to
1345 % bail out on encountering a space or an opening brace.
1346 \cs_new:Npn \um_peek_execute_branches_ss: {
1347   \bool_if:nTF {
1348     \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
1349     \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
1350   } { \l_peek_false_tl }
1351   { \um_peek_execute_branches_ss_aux: }
1352 }
1353
1354 % This is the actual comparison code.

```

```

1355 % Because the peeking has already tokenised the next token,
1356 % it's too late to extract its charcode directly. Instead,
1357 % we look at its meaning, which remains a 'character' even
1358 % though it is itself math-active. If the character is ever
1359 % made fully active, this will break our assumptions!
1360 %
1361 % If the char's meaning exists as a property list key, we
1362 % build up a chain of sub-/superscripts and iterate. (If not, exit and
1363 % typeset what we've already collected.)
1364 \cs_new:Nn \um_peek_execute_branches_ss_aux: {
1365   \prop_if_in:cxTF
1366     {g_um_\l_um_tmpa_tl _prop}
1367     {\meaning\l_peek_token}
1368   {
1369     \prop_get:cxN
1370       {g_um_\l_um_tmpa_tl _prop}
1371       {\meaning\l_peek_token}
1372       \l_um_tmpb_tl
1373       \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmpb_tl
1374       \l_peek_true_tl
1375   }
1376   {\l_peek_false_tl}
1377 }

```

9.0.4 Compatibility

Just comment out the offending lines for now:

```

1378 \@ifpackageloaded{amsopn}{
1379   \cs_set:Npn \newmcodes@ {
1380     \mathcode'\ '39
1381     \mathcode'\ *42
1382     \mathcode'\ ."613A%
1383   % \ifnum\mathcode'\-=45 \else
1384   %   \mathchardef\std@minus\mathcode'\-\relax
1385   % \fi
1386     \mathcode'\-45
1387     \mathcode'\ /47
1388     \mathcode'\ : "603A\relax
1389   }
1390 }{}

```

9.0.5 Synonyms and all the rest

We need to change L^AT_EX's idea of the font used to typeset things like `\sin` and `\cos`:

```

1391 \def\operator@font{\um_setup_mathup:}

```

```

1392 \def\to{\rightarrow}
1393 \def\le{\leq}
1394 \def\ge{\geq}

\mathcal
1395 \def\mathcal{\mathscr}

\mathrm
1396 \def\mathrm{\mathup}

    Overriding amsmath definitions:
1397 \AtBeginDocument{
1398   \def\@cdots{\mathinner{\cdots}}
1399 }

    Interaction with beamer:
1400 \AtBeginDocument{
1401   \ifpackageloaded{beamer}{
1402     \ifbeamer@suppressreplacements\else
1403       \PackageWarningNoLine{unicode-math}{
1404         Disabling~ beamer's~ math~ setup.^J
1405         Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1406       }
1407       \beamer@suppressreplacementstrue
1408     \fi
1409   }{}
1410 }

    The end.
1411 \ExplSyntaxOff

```

File II

STIX table data extraction

The source for the \TeX names for the very large number of mathematical glyphs are provided via Barbara Beeton's table file for the STIX project (ams.org/STIX). A version is located at <http://www.ams.org/STIX/bnb/stix-tbl.asc> but check <http://www.ams.org/STIX/> for more up-to-date info.

This table is converted into a form suitable for reading by \XeTeX , and then hand-edited by the author; the result is `unicode-math-table.tex`.

A single file is produced containing all (more than 3298) symbols. Future optimisations might include generating various (possibly overlapping) subsets so not all definitions must be read just to redefine a small range of symbols. Performance for now seems to be acceptable without such measures.

```
1 #!/bin/sh
2
3 cat stix-tbl.txt |
4 awk '
```

If the USV isn't repeated (TODO: check this is valid!) and the entry isn't one of the weird ones in the big block at the end of the STIX table (TODO: check that out!)...

```
5 {if (usv != substr($0,2,5) && substr($0,2,1) != " ")
6 {usv = substr($0,2,5);
7   texname = substr($0,84,25);
8   class = substr($0,57,1);
9   description = tolower(substr($0,233,350));
```

If the USV has a macro name, which isn't `\text...`, and isn't a single character macro (e.g., `\#`, `\S`, ...), and has a class, and it isn't reserved (*i.e.*, doubled up with a previously assigned glyph):

```

10     if (texname      ~ /[\\]/ &&
11         substr(texname,0,5) != "\\text"      &&
12         substr(texname,0,4) != "\\ipa"      &&
13         substr(texname,0,5) != "\\tone"     &&
14         substr(texname,3,1) != " "          &&
15         class         != " "                &&
16         description !~ /<reserved>/ )

```

Print the actual entry corresponding to the unicode character:

```

17     print "\\UnicodeMathSymbol{"\" \" \" \
18         usv "}" \" \" \
19         texname "}" \" \" \
20         class "}" \" \" \
21         description "%}";
22 }' - |

```

Now replace the `stix` class abbreviations with their \TeX macro names.

```
23 sed -e ' s/{N}/{\\mathord}/ ' \
```

A ‘fence’ defined by the `stix` table is something like `\vert`; in `XYTeX` this is just a `\mathord` that will grow with the magic of `\XeTeXmathchardef`.

```

24 -e ' s/{F}/{\\mathord}/ ' \
25 -e ' s/{A}/{\\mathalpha}/ ' \
26 -e ' s/{D}/{\\mathaccent}/ ' \
27 -e ' s/{P}/{\\mathpunct}/ ' \
28 -e ' s/{B}/{\\mathbin}/ ' \
29 -e ' s/{R}/{\\mathrel}/ ' \
30 -e ' s/{L}/{\\mathop}/ ' \
31 -e ' s/{O}/{\\mathopen}/ ' \
32 -e ' s/{C}/{\\mathclose}/ ' \

```

Fixing up a couple of things in the STIX table.

```
33 -e ' s/\^/\\string^/ ' > unicode-math.tex
```


A Documenting maths support in the NFSS

A.1 Overview

In the following, $\langle NFSS\ decl.\rangle$ stands for something like $\{\mathrm{T1}\}\{\mathrm{lmr}\}\{\mathrm{m}\}\{\mathrm{n}\}$.

Maths symbol fonts Fonts for symbols: $\alpha, \leq, \rightarrow$

`\DeclareSymbolFont{<name>}{NFSS decl.}`

Declares a named maths font such as operators from which symbols are defined with `\DeclareMathSymbol`.

Maths alphabet fonts Fonts for $ABC-xyz, \mathfrak{ABC}-\mathcal{XYZ}$, etc.

`\DeclareMathAlphabet{<cmd>}{NFSS decl.}`

For commands such as `\mathbf`, accessed through maths mode that are unaffected by the current text font, and which are used for alphabetic symbols in the ASCII range.

`\DeclareSymbolFontAlphabet{<cmd>}{<name>}`

Alternative (and optimisation) for `\DeclareMathAlphabet` if a single font is being used for both alphabetic characters (as above) and symbols.

Maths ‘versions’ Different maths weights can be defined with the following, switched in text with the `\mathversion{<maths version>}` command.

`\SetSymbolFont{<name>}{<maths version>}{NFSS decl.}`

`\SetMathAlphabet{<cmd>}{<maths version>}{NFSS decl.}`

Maths symbols Symbol definitions in maths for both characters (=) and macros (`\eqdef`): `\DeclareMathSymbol{<symbol>}{<type>}{<named font>}{<slot>}` This is the macro that actually defines which font each symbol comes from and how they behave.

Delimiters and radicals use wrappers around $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ’s `\delimiter`/`\radical` primitives, which are re-designed in $\mathrm{X}_{\mathrm{E}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$. The syntax used in $\mathrm{L}_{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ’s NFSS is therefore not so relevant here.

Delimiters A special class of maths symbol which enlarge themselves in certain contexts.

`\DeclareMathDelimiter{<symbol>}{<type>}{<sym. font>}{<slot>}{<sym. font>}{<slot>}`

Radicals Similar to delimiters (`\DeclareMathRadical` takes the same syntax) but behave ‘weirdly’. `\sqrt` might very well be the only one.

In those cases, glyph slots in *two* symbol fonts are required; one for the small (‘regular’) case, the other for situations when the glyph is larger. This is not the case in $\mathrm{X}_{\mathrm{E}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$.

Accents are not included yet.

Summary For symbols, something like:

```
\def\DeclareMathSymbol#1#2#3#4{
  \global\mathchardef#1"\mathchar@type#2
  \expandafter\hexnumber@\csname sym#2\endcsname
  {\hexnumber@{\count\z@}\hexnumber@{\count\tw@}}}
```

For characters, something like:

```
\def\DeclareMathSymbol#1#2#3#4{
  \global\mathcode`#1"\mathchar@type#2
  \expandafter\hexnumber@\csname sym#2\endcsname
  {\hexnumber@{\count\z@}\hexnumber@{\count\tw@}}}
```

File III

X_YTeX math font dimensions

These are the extended `\fontdimens` available for suitable fonts in X_YTeX. Note that LuaTeX takes an alternative route, and this package will eventually provide a wrapper interface to the two (I hope).

<code>\fontdimen</code>	Dimension name	Description
10	<code>SCRIPTPERCENTSCALEDOWN</code>	Percentage of scaling down for script level 1. Suggested value: 80%.
11	<code>SCRIPTSCRIPTPERCENTSCALEDOWN</code>	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	<code>DELIMITEDSUBFORMULAMINHEIGHT</code>	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height \times 1.5.
13	<code>DISPLAYOPERATORMINHEIGHT</code>	Minimum height of n-ary operators (such as integral and summation) for formulas in display mode.
14	<code>MATHLEADING</code>	White space to be left between math formulas to ensure proper line spacing. For example, for applications that treat line gap as a part of line ascender, formulas with ink going above (<code>os2.sTypoAscender</code> + <code>os2.sTypoLineGap</code> – <code>MathLeading</code>) or with ink going below <code>os2.sTypoDescender</code> will result in increasing line height.

\fontdimen	Dimension name	Description
15	AxisHEIGHT	Axis height of the font.
16	ACCENTBASEHEIGHT	Maximum (ink) height of accent base that does not require raising the accents. Suggested: x-height of the font (os2.sxHeight) plus any possible overshots.
17	FLATTENEDACCENTBASE-HEIGHT	Maximum (ink) height of accent base that does not require flattening the accents. Suggested: cap height of the font (os2.sCapHeight).
18	SUBSCRIPTSHIFTDOWN	The standard shift down applied to subscript elements. Positive for moving in the downward direction. Suggested: os2.ySubscriptYOffset.
19	SUBSCRIPTTOPMAX	Maximum allowed height of the (ink) top of subscripts that does not require moving subscripts further down. Suggested: $\frac{1}{5}$ x-height.
20	SUBSCRIPTBASELINEDROPMIN	Minimum allowed drop of the baseline of subscripts relative to the (ink) bottom of the base. Checked for bases that are treated as a box or extended shape. Positive for subscript baseline dropped below the base bottom.
21	SUPERSCRIPSHIFTUP	Standard shift up applied to superscript elements. Suggested: os2.ySuperscriptYOffset.
22	SUPERSCRIPSHIFTUPCRAMPED	Standard shift of superscripts relative to the base, in cramped style.
23	SUPERSCRIPBOTTOMMIN	Minimum allowed height of the (ink) bottom of superscripts that does not require moving subscripts further up. Suggested: $\frac{1}{4}$ x-height.
24	SUPERSCRIPBASELINEDROP-MAX	Maximum allowed drop of the baseline of superscripts relative to the (ink) top of the base. Checked for bases that are treated as a box or extended shape. Positive for superscript baseline below the base top.
25	SUBSUPERSCRIPGAPMIN	Minimum gap between the superscript and subscript ink. Suggested: 4×default rule thickness.

\fontdimen	Dimension name	Description
26	SUPERSCRIPBTOTTOMMAX- WITHSUBSCRIPT	The maximum level to which the (ink) bottom of superscript can be pushed to increase the gap between superscript and subscript, before subscript starts being moved down. Suggested: $\frac{1}{5}$ x-height.
27	SPACEAFTERSRIPT	Extra white space to be added after each subscript and superscript. Suggested: 0.5pt for a 12 pt font.
28	UPPERLIMITGAPMIN	Minimum gap between the (ink) bottom of the upper limit, and the (ink) top of the base operator.
29	UPPERLIMITBASELINERISEMIN	Minimum distance between baseline of upper limit and (ink) top of the base operator.
30	LOWERLIMITGAPMIN	Minimum gap between (ink) top of the lower limit, and (ink) bottom of the base operator.
31	LOWERLIMITBASELINEDROP- MIN	Minimum distance between baseline of the lower limit and (ink) bottom of the base operator.
32	STACKTOPSHIFTUP	Standard shift up applied to the top element of a stack.
33	STACKTOPDISPLAYSTYLESHIFT- UP	Standard shift up applied to the top element of a stack in display style.
34	STACKBOTTOMSHIFTDOWN	Standard shift down applied to the bottom element of a stack. Positive for moving in the downward direction.
35	STACKBOTTOMDISPLAYSTYLE- SHIFTDOWN	Standard shift down applied to the bottom element of a stack in display style. Positive for moving in the downward direction.
36	STACKGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element. Suggested: $3 \times$ default rule thickness.
37	STACKDISPLAYSTYLEGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element in display style. Suggested: $7 \times$ default rule thickness.
38	STRETCHSTACKTOPSHIFTUP	Standard shift up applied to the top element of the stretch stack.

\fontdimen	Dimension name	Description
39	STRETCHSTACKBOTTOMSHIFT-DOWN	Standard shift down applied to the bottom element of the stretch stack. Positive for moving in the downward direction.
40	STRETCHSTACKGAPABOVEMIN	Minimum gap between the ink of the stretched element, and the (ink) bottom of the element above. Suggested: UpperLimitGapMin
41	STRETCHSTACKGAPBELOWMIN	Minimum gap between the ink of the stretched element, and the (ink) top of the element below. Suggested: LowerLimitGapMin.
42	FRACTIONNUMERATORSHIFTUP	Standard shift up applied to the numerator.
43	FRACTIONNUMERATOR-DISPLAYSTYLESHIFTUP	Standard shift up applied to the numerator in display style. Suggested: StackTopDisplayStyleShiftUp.
44	FRACTIONDENOMINATORSHIFT-DOWN	Standard shift down applied to the denominator. Positive for moving in the downward direction.
45	FRACTIONDENOMINATOR-DISPLAYSTYLESHIFTDOWN	Standard shift down applied to the denominator in display style. Positive for moving in the downward direction. Suggested: StackBottomDisplayStyleShiftDown.
46	FRACTIONNUMERATORGAP-MIN	Minimum tolerated gap between the (ink) bottom of the numerator and the ink of the fraction bar. Suggested: default rule thickness
47	FRACTIONNUMDISPLAYSTYLE-GAPMIN	Minimum tolerated gap between the (ink) bottom of the numerator and the ink of the fraction bar in display style. Suggested: 3×default rule thickness.
48	FRACTIONRULETHICKNESS	Thickness of the fraction bar. Suggested: default rule thickness.
49	FRACTIONDENOMINATORGAP-MIN	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar. Suggested: default rule thickness

\fontdimen	Dimension name	Description
50	FRACTIONDENOMDISPLAY- STYLEGAPMIN	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar in display style. Suggested: 3×default rule thickness.
51	SKEWEDFRACTION- HORIZONTALGAP	Horizontal distance between the top and bottom elements of a skewed fraction.
52	SKEWEDFRACTIONVERTICAL- GAP	Vertical distance between the ink of the top and bottom elements of a skewed fraction.
53	OVERBARVERTICALGAP	Distance between the overbar and the (ink) top of the base. Suggested: 3×default rule thickness.
54	OVERBARRULETHICKNESS	Thickness of overbar. Suggested: default rule thickness.
55	OVERBAREXTRAASCENDER	Extra white space reserved above the overbar. Suggested: default rule thickness.
56	UNDERBARVERTICALGAP	Distance between underbar and (ink) bottom of the base. Suggested: 3×default rule thickness.
57	UNDERBARRULETHICKNESS	Thickness of underbar. Suggested: default rule thickness.
58	UNDERBAREXTRADESCENDER	Extra white space reserved below the underbar. Always positive. Suggested: default rule thickness.
59	RADICALVERTICALGAP	Space between the (ink) top of the expression and the bar over it. Suggested: 1¼ default rule thickness.
60	RADICALDISPLAYSTYLE- VERTICALGAP	Space between the (ink) top of the expression and the bar over it. Suggested: default rule thickness + ¼ x-height.
61	RADICALRULETHICKNESS	Thickness of the radical rule. This is the thickness of the rule in designed or constructed radical signs. Suggested: default rule thickness.
62	RADICALEXTRAASCENDER	Extra white space reserved above the radical. Suggested: RadicalRuleThickness.
63	RADICALKERNBEFOREDEGREE	Extra horizontal kern before the degree of a radical, if such is present. Suggested: 5/18 of em.

\fontdimen	Dimension name	Description
64	RADICALKERNAFTERDEGREE	Negative kern after the degree of a radical, if such is present. Suggested: $-10/18$ of em.
65	RADICALDEGREEBOTTOM-RAISEPERCENT	Height of the bottom of the radical degree, if such is present, in proportion to the ascender of the radical sign. Suggested: 60%.

File IV

Some manner of unit testing

Some of the examples in the documentation are actually set up as unit tests, where multiple maths alphabets are placed on top of each other to ensure that various input methods result in the same output.

B The regular weight alphabets

For regular weight alphabets, we test the resolution from upright/italic math source to unified-shape output.

```

1 (*test)
2 \documentclass{article}
3 \usepackage[a6paper]{geometry}
4 \usepackage{fontspec}
5 \setmainfont{FPL Neu}
6 \usepackage{unicode-math}
7 \def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
8 \def\uplatin{abcdefghijklmnopqrstuvwxyz}
9 \def\upGreek{
10 \def\upgreek{
11 \def\itLatin{
12 \def\itlatin{
13 \def\itGreek{
14 \def\itgreek{
15 \def\testmath#1{%
16   \makebox[\linewidth][l]{%
17     \makebox[0pt][l]{\csname up#1\endcsname$}%
18     \makebox[0pt][l]{\csname it#1\endcsname$}}
19 \begin{document}
20 \setmathfont[Colour=2255FF99]{Asana Math}
21 \parindent=0pt
22 \voffset=-1in

```

```

23 \hoffset=-1in
24 \setbox0=\vbox{%
25 \testmath{Latin}\\
26 \testmath{latin}\\
27 \testmath{Greek}\\
28 \testmath{greek}}
29 \dimen0=\ht0
30 \advance\dimen0\dp0
31 \edef\papersize{papersize=\the\wd0,\the\dimen0}
32 \setbox255=\vbox{\special{\papersize}\box0}
33 \shipout\box255
34 \end{document}
35 </test>

```

We need three unit tests to produce the three variations of the math-style option. I'm guessing `literal` is working just fine, but it really needs a different test.

C The bold alphabets

For bold alphabets, it's a bit more complex. We also test `literal bold` to the bold produced from markup.

```

36 <*testbf>
37 \documentclass{article}
38 \usepackage[a6paper]{geometry}
39 \usepackage{fontspec}
40 \setmainfont{FPL Neu}
41 \usepackage{unicode-math}
42 \def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
43 \def\uplatin{abcdefghijklmnopqrstuvwxyz}
44 \def\upGreek{
45 \def\upgreek{
46 \def\itLatin{
47 \def\itlatin{
48 \def\itGreek{
49 \def\itgreek{
50 \def\bfupLatin{
51 \def\bfuplatin{
52 \def\bfupGreek{
53 \def\bfupgreek{
54 \def\bfitalicLatin{
55 \def\bfitaliclatin{
56 \def\bfitalicGreek{
57 \def\bfitalicgreek{
58 \providecommand\mathalphabet{\mathbf}

```



```

59 \def\testmath#1{%
60   \makebox[\linewidth][l]{%
61     \makebox[0pt][l]{\mathalphabet{\csname up#1\endcsname}$}%
62     \makebox[0pt][l]{\mathalphabet{\csname it#1\endcsname}$}%
63     \makebox[0pt][l]{\csname bfup#1\endcsname$}%
64     \makebox[0pt][l]{\csname bfit#1\endcsname$}%
65   }}
66 \begin{document}
67 \setmathfont[Colour=2255FF55]{Asana Math}
68 \parindent=0pt
69 \voffset=-1in
70 \hoffset=-1in
71 \setbox0=\vbox{%
72   \testmath{Latin}\\
73   \testmath{latin}\\
74   \testmath{Greek}\\
75   \testmath{greek}}
76 \dimen0=\ht0
77 \advance\dimen0\dp0
78 \edef\papersize{papersize=\the\wd0,\the\dimen0}
79 \setbox255=\vbox{\special{\papersize}\box0}
80 \shipout\box255
81 \end{document}
82 </testbf>

```

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols	
\"	17
\'	1211
\-	9
\:::	674
\::f	674
\::n	674
\=	10
\DeclareMathDelimiter	276
\DeclareMathSizes	241
\backslashchar	838
\begindocumenthook	8
\ecclvi	301
\cdots	1381
\elt	545–549, 552, 556, 558
\empty	352, 353, 390, 473, 590, 597, 612, 633, 638
\ifnextchar	1216
\ifpackageloaded	7, 1384
\ii	596, 597, 599, 601, 604, 609
\input	407, 1162
\marker	609, 628
\nil	310, 454, 609, 622–625, 1143, 1148, 1160
\preamblecmds	285
\sqrt	1216
\tempa	125, 168, 191, 209, 227, 363, 379, 588, 600, 624, 626, 638
\tempb	125, 126, 168, 169, 191, 192, 209, 210, 227, 228, 588, 589, 627, 628, 633
\tempswafalse	598
\tempswatru	602, 605, 630, 635, 640, 645
\um@bfliteraltrue	164, 188
\um@bfupGreekfalse	131, 170
\um@bfupGreektrue	143, 155, 176, 182
\um@bfupLatinfalse	134, 173
\um@bfupLatintrue	146, 158, 179, 185
\um@bfupgreekfalse	132, 144, 171, 177
\um@bfupgreektrue	156, 183
\um@bfuplatinfalse	135, 174
\um@bfuplatintrue	147, 159, 180, 186
\um@fontspec@featuretrue	354
\um@literaltrue	163
\um@ot@math@true	382
\um@upGreekfalse	127
\um@upGreektrue	139, 151
\um@upLatinfalse	129, 141
\um@upLatintrue	153
\um@upNablafalse	136, 195
\um@upNabltrue	148, 160, 193
\um@upgreekfalse	128, 140
\um@upgreektrue	152
\um@uplatinfalse	130, 142, 154
\um@uppartialfalse	133, 145, 172, 178, 213
\um@uppartialtrue	157, 184, 211
\xDeclareMathDelimiter	277
\xxDeclareMathDelimiter	275
\	10–13, 17, 23–33, 72–74
\^	33, 1142, 1151
Numbers	
\0	29
_	17–20, 23–32
A	
\A	30
\a	31
\addnolimits	<u>551</u>
\addtoversion	248
\advance	30, 77
\alloc@	301
\Alpha	701
\alpha	726
\alpha@elt	252
\alpha@list	251
\AtBeginDocument	699, 1380, 1383
\awint	547

B	
\B	929
\beamer@suppressreplacementstrue	1390
\begin	19, 66
\begingroup	307, 555, 856, 1141, 1158
\Beta	702
\beta	727, 1312
\bfitGreek	56
\bfitgreek	57
\bfitLatin	54
\bfitlatin	55
\bfupGreek	52
\bfupgreek	53
\bfupLatin	50
\bfuplatin	51
\bgroup	1188
\bool_if:NTF	731, 750, 756, 761
\bool_if:nTF	1342
\bool_new:N	28
\bool_set_false:N	137, 161, 165, 229
\bool_set_true:N	149, 231
\box	32, 33, 79, 80, 1226
C	
\C	918, 944
\c_group_begin_token	1343
\c_peek_true_remove_next_tl	1333
\c_space_token	1344
\cdots	1381
\cdp@elt	239
\cdp@list	238
\char_make_active:N	1211
\char_make_active:n	308, 1212, 1244, 1277
\char_make_math_superscript:N ..	1151
\char_make_other:N	1142
\chardef	301
\Chi	723
\chi	752, 1316
\cirfnint	547
\clist_map_inline:Nn	530
\clist_map_inline:nn	459, 525, 532
\clist_map_variable:NNn	596
\clist_map_variable:nNn .	653, 676, 682
\copy	1224
\cs	1240, 1329
\cs_generate_variant:Nn	1230–1232
\cs_gset:cpn	315, 325
\cs_gset:Npn ..	328, 333, 1143, 1247, 1280
\cs_gset:Npx	338
\cs_gset_eq:NN	1213, 1214
\cs_if_exist:cF	854
\cs_if_exist:cT	839
\cs_new:Nn	414, 426, 453, 458, 463, 466, 675, 681, 690, 693, 696, 831, 853, 866, 1167, 1171, 1187, 1192, 1321, 1331, 1360
\cs_new:Npn ..	884, 899, 915, 927, 942, 951, 956, 961, 966, 1043, 1063, 1083, 1088, 1093, 1109, 1125, 1341
\cs_set:cpn	855
\cs_set:Nn	198, 216, 347, 418, 421, 472, 524, 529, 652, 662, 665, 668, 671, 766, 777, 787, 794, 803, 812, 844, 847, 1242, 1275
\cs_set:Npn	674
\cs_set_eq:cN	848
\cs_set_eq:NN	393–396, 400–403, 1209, 1249, 1282
\cs_set_protected:cpn	858
\cs_to_str:N	325, 540
\csname	17, 18, 61–64, 304, 310, 313, 316, 348, 355, 468, 540, 832, 860
D	
\DeclareDocumentCommand	350
\DeclareMathAccent	269
\DeclareMathAlphabet	265
\DeclareMathDelimiter	274
\DeclareMathRadical	280
\DeclareMathSizes	240
\DeclareMathSymbol	271
\DeclareMathVersion	245, 357
\DeclareRobustCommand	1216
\DeclareSymbolFont	258, 405
\DeclareSymbolFontAlphabet	282
\DeclareSymbolFontAlphabet@	283
\def	7–15, 29–69, 71–123, 287, 290, 301, 303, 305, 356, 544, 552, 554, 556, 563, 565, 622, 624–627, 701–730, 733–749, 752–755, 758–760, 763, 764, 1153, 1159, 1217, 1374–1379, 1381
\define@choicekey	124, 168, 191, 209, 227, 588

\define@cmdkey	584–587	\f@size	355
\define@key	567	\fi ..	166, 189, 196, 207, 214, 225, 232, 298, 299, 320, 341–345, 389, 404, 436, 451, 482, 490, 495, 502, 519, 520, 522, 535, 559, 575, 591, 606, 607, 610, 616, 618, 619, 631, 636, 641, 646–650, 792, 801, 810, 829, 1005, 1012, 1019, 1038, 1041, 1391
\define@mathalphabet	246	\fi:	851
\define@mathgroup	247	\fint	547
\Delta	704	\font	380
\delta	729	\fontdimen	288, 381, 1219, 1225
\dimen	29–31, 76–78	\fontname	838
\dimexpr	288, 381, 1220		
\do	285		
\documentclass	2, 37		
\dorestore@version	255		
\dp	30, 77, 1222		
E		G	
\E	930	\g	939
\e	938	\g@addto@macro	539, 613, 615
\edef	31, 78, 363, 623, 624	\g_um_primekern_muskip	1164, 1165, 1169
\egroup	1204	\g_um_subs_prop	1229, 1276
\else	203, 221, 293, 296, 322, 326, 331, 336, 339, 383, 397, 433, 446, 477, 485, 488, 493, 499, 511, 521, 557, 570, 603, 608, 614, 632, 637, 642, 790, 798, 807, 821, 1000, 1003, 1009, 1016, 1029, 1385	\g_um_supers_prop	1228, 1243
\else:	851	\g_um_texgreek_bool ..	28, 137, 149, 161, 165, 229, 231, 731, 750, 756, 761
\encodingdefault	406, 845	\Gamma	703
\end	34, 81	\gamma	728, 1313
\endcsname ...	17, 18, 61–64, 304, 310, 313, 316, 348, 355, 468, 540, 832, 860	\gdef	1148
\endgroup	311, 561, 856, 1157, 1163	\ge	1377
\Epsilon	705	\geq	1377
\epsilon	730	\get@cdp	264
\Eta	707	\glb@currsiz	351
\eta	734	\global	309, 312, 329, 330, 334, 335, 340, 590, 1145, 1153, 1245, 1278
\etex_iffontchar:D	851	\group@elt	260
\ExecuteOptionsX	234	\group@list	259
\exp_args:Nnff	674, 677, 684	\group_align_safe_begin:	1335
\exp_args:No	526, 533	\group_align_safe_end:	1334
\exp_not:n	1332, 1334	\group_begin:	1210, 1234
\expandafter	304, 312, 317, 319, 323, 539, 552, 599, 601, 604, 609, 623, 624, 628, 860	\group_end:	1215, 1318
\ExplSyntaxOff	1394		
\ExplSyntaxOn	6		
F		H	
\F	931	\H	919, 932, 945
		\h	797, 800, 902
		\hbox	1218
		\hoffset	23, 70
		\ht	29, 76, 1221
I		I	
		\I	933, 946

\backslash if@tempswa	611	L	\backslash L	934
\backslash if@um@bfliteral	21, 437, 483, 970	\backslash l_peek_false_tl	1334, 1346, 1372	
\backslash if@um@bfupGreek	22, 496, 1013	\backslash l_peek_token	1343, 1344, 1363, 1367	
\backslash if@um@bfupgreek	23, 503, 1020	\backslash l_peek_true_aux_tl	1332	
\backslash if@um@bfupLatin	24, 486, 1001	\backslash l_peek_true_tl	1333, 1370	
\backslash if@um@bfuplatin	25, 491, 1006	\backslash l_um_inc_num	683, 685, 686	
\backslash if@um@fontspec@feature	14, 568	\backslash l_um_incr_num	654, 656, 658	
\backslash if@um@literal	16, 428, 475	\backslash l_um_input_num	653, 656, 676, 678, 682, 685	
\backslash if@um@ot@math@	15	\backslash l_um_primecount_num	1166, 1189, 1193, 1203	
\backslash if@um@upGreek	17, 804	\backslash l_um_script_features_tl	358, 369	
\backslash if@um@upgreek	18, 813	\backslash l_um_script_font_tl	360, 368	
\backslash if@um@upLatin	19, 788	\backslash l_um_ss_chain_tl	1248, 1281, 1325, 1369	
\backslash if@um@uplatin	20, 795	\backslash l_um_sscript_features_tl	359, 373	
\backslash if@um@upNabla	26, 199	\backslash l_um_sscript_font_tl	361, 372	
\backslash if@um@uppartial	27, 217	\backslash l_um_tmpa_tl	1250, 1283, 1362, 1366	
\backslash ifbeamer@suppressreplacements	1385	\backslash l_um_tmpb_tl	1368, 1369	
\backslash ifcase	126, 169, 192, 210, 228, 589	\backslash Lambda	711	
\backslash ifdim	381	\backslash lambda	738	
\backslash ifin@	318, 324	\backslash le	1376	
\backslash ifnum	531, 629, 634, 639, 643, 644	\backslash left	564	
\backslash ifx	291, 294, 306, 327, 332, 337, 390, 473, 557, 597, 600, 601, 604, 612, 628, 633, 638	\backslash left@primitive	564, 565	
\backslash iiiint	545	\backslash leq	1376	
\backslash iiint	545	\backslash let	56, 70, 302, 351–353, 564, 590, 1145	
\backslash iint	545	\backslash linewidth	16, 60	
\backslash in@	317, 323	\backslash lowercase	1144, 1149	
\backslash init@restore@version	254	\backslash lowint	549	
\backslash int	545	M		
\backslash intBar	547	\backslash M	935	
\backslash intbar	547	\backslash m@th	1218	
\backslash intcap	549	\backslash makebox	16–18, 60–64	
\backslash intclockwise	546	\backslash mathaccent	337	
\backslash intcup	549	\backslash mathalpha	541, 657	
\backslash intlarhk	548	\backslash mathalphabet	58, 61, 62	
\backslash intx	548	\backslash mathbb	916–925	
\backslash Iota	709	\backslash mathbf	58, 967–969, 971–999, 1002, 1004, 1007, 1008, 1010, 1011, 1014, 1015, 1017, 1018, 1021–1028, 1030–1037, 1039, 1040	
\backslash iota	736	\backslash mathbffrak	1084–1086	
\backslash itGreek	13, 48	\backslash mathbfit	1044–1061	
\backslash itgreek	14, 49	\backslash mathbfscr	1089–1091	
\backslash itLatin	11, 46	\backslash mathbfsf	1094–1107	
\backslash itlatin	12, 47	\backslash mathbfsfit	1126–1139	
		\backslash mathbfsfup	1110–1123	
K				
\backslash Kappa	710			
\backslash kappa	737			
\backslash kern	1219, 1225			

<code>\mathbfup</code>	1064–1081	<code>\mitNu</code>	713
<code>\mathbin</code>	427	<code>\mitnu</code>	740
<code>\mathcal</code>	1378	<code>\mitOmega</code>	725
<code>\mathchar@type</code>	281, 314, 328, 330, 333, 335, 338, 340, 348, 467	<code>\mitomega</code>	754
<code>\mathchardef</code>	9, 10	<code>\mitOmicron</code>	715
<code>\mathclose</code>	332	<code>\mitomicron</code>	742
<code>\mathcode</code>	9, 10, 309	<code>\mitPhi</code>	722
<code>\mathfrak</code>	943–949	<code>\mitphi</code>	750, 761
<code>\mathgroup</code>	301	<code>\mitPi</code>	716
<code>\mathinner</code>	1381	<code>\mitpi</code>	743
<code>\mathit</code>	900–913	<code>\mitPsi</code>	724
<code>\mathop</code>	306	<code>\mitpsi</code>	753
<code>\mathopen</code>	327, 565	<code>\mitRho</code>	717
<code>\mathord</code>	429–432, 434, 435, 438–445, 447–450, 464	<code>\mitrho</code>	744
<code>\mathrm</code>	1379	<code>\mitSigma</code>	719
<code>\mathscr</code>	928–940, 1378	<code>\mitsigma</code>	746
<code>\mathsf</code>	952–954	<code>\mitTau</code>	720
<code>\mathsf{fit}</code>	957–959	<code>\mittau</code>	747
<code>\mathtt</code>	845, 962–964	<code>\mitTheta</code>	708
<code>\mathup</code>	885–897, 1379	<code>\mittheta</code>	735
<code>\mddefault</code>	406, 845	<code>\mitUpsilon</code>	721
<code>\meaning</code>	1243, 1276, 1363, 1367	<code>\mitupsilon</code>	748
<code>\mitAlpha</code>	701	<code>\mitvarepsilon</code>	731, 756
<code>\mitalpha</code>	726	<code>\mitvarkappa</code>	759
<code>\mitBeta</code>	702	<code>\mitvarphi</code>	750, 761
<code>\mitbeta</code>	727	<code>\mitvarpi</code>	764
<code>\mitChi</code>	723	<code>\mitvarrho</code>	763
<code>\mitchi</code>	752	<code>\mitvarsigma</code>	745
<code>\mitDelta</code>	704	<code>\mitvarTheta</code>	718
<code>\mitdelta</code>	729	<code>\mitvartheta</code>	758
<code>\mitEpsilon</code>	705	<code>\mitXi</code>	714
<code>\mitepsilon</code>	731, 756	<code>\mitxi</code>	741
<code>\mitEta</code>	707	<code>\mitZeta</code>	706
<code>\miteta</code>	734	<code>\mitzeta</code>	733
<code>\mitGamma</code>	703	<code>\mode_if_math:F</code>	859
<code>\mitgamma</code>	728	<code>\mskip</code>	1169
<code>\mitIota</code>	709	<code>\Mu</code>	712
<code>\mitiota</code>	736	<code>\mu</code>	739
<code>\mitKappa</code>	710	<code>\muskip_gset:Nn</code>	1165
<code>\mitkappa</code>	737	<code>\muskip_new:N</code>	1164
<code>\mitLambda</code>	711		
<code>\mitlambda</code>	738		
<code>\mitMu</code>	712		
<code>\mitmu</code>	739		

<code>\newcommand</code>	538, 551, 566, 595, 700	<code>\pi</code>	743
<code>\newcounter</code>	13	<code>\pointint</code>	548
<code>\newfam</code>	302	<code>\prg_case_int:nnn</code>	1172
<code>\newif</code>	14–27	<code>\prg_do_nothing:</code>	848
<code>\newmathalphabet</code>	242	<code>\prg_new_conditional:Nnn</code>	850
<code>\newmathalphabet@@</code>	243	<code>\prg_replicate:nn</code>	1169
<code>\newmathalphabet@@@</code>	244	<code>\prg_return_false:</code>	851
<code>\noexpand</code>	363, 558	<code>\prg_return_true:</code>	851
<code>\nolimits</code>	319	<code>\prg_stepwise_variable:nnnNn</code>	654, 683
<code>\non@alpherr</code>	860	<code>\prime</code>	1209
<code>\npolint</code>	548	<code>\primedouble</code>	1175
<code>\Nu</code>	713	<code>\primequadruple</code>	1181
<code>\nu</code>	740	<code>\primesingle</code>	464, 1168, 1169, 1173
<code>\num_incr:N</code>	1193	<code>\primetripel</code>	1178
<code>\num_new:N</code>	1166	<code>\process@table</code>	256
<code>\num_zero:N</code>	1189	<code>\ProcessOptionsX</code>	235
<code>\number</code>	678, 685, 686	<code>\prop_get:cxN</code>	1365
<code>\numexpr</code>	634,	<code>\prop_get:NnN</code>	1231
	639, 643, 644, 656, 658, 678, 685, 686	<code>\prop_gput:Nnn</code>	1230
O		<code>\prop_gput:Nxn</code>	1243, 1276
<code>\o</code>	940	<code>\prop_if_in:cxTF</code>	1361
<code>\oiint</code>	545	<code>\prop_if_in:NnTF</code>	1232
<code>\oint</code>	545	<code>\prop_new:N</code>	1228, 1229
<code>\ointctrclockwise</code>	546	<code>\protect</code>	574
<code>\Omega</code>	725	<code>\providecommand</code>	58
<code>\omega</code>	754	<code>\ProvidesPackage</code>	1
<code>\Omicron</code>	715	<code>\Psi</code>	724
<code>\omicron</code>	742	<code>\psi</code>	753
<code>\operator@font</code>	1374	Q	
<code>\or</code>	138,	<code>\Q</code>	922
	150, 162, 175, 181, 187, 194, 212, 230	R	
P		<code>\R</code>	923, 936, 947
<code>\P</code>	921	<code>\r@t</code>	<u>1217</u>
<code>\PackageError</code>	571	<code>\raise</code>	1220
<code>\PackageInfo</code>	392	<code>\relax</code>	9, 10, 126,
<code>\PackageWarningNoLine</code>	384, 837, 1386		169, 192, 210, 228, 288, 306, 309,
<code>\papersize</code>	31, 32, 78, 79		314, 325, 327–330, 332–335, 337,
<code>\parindent</code>	21, 68		338, 340, 348, 351, 380, 381, 531,
<code>\peek_after:NN</code>	1336		589, 601, 604, 628, 629, 634, 639,
<code>\peek_charcode_remove:NTF</code>	1194, 1200		643, 644, 656, 658, 678, 685, 686, 1223
<code>\peek_meaning_remove:NTF</code>	1197	<code>\removenolimits</code>	<u>554</u>
<code>\Phi</code>	722	<code>\RequirePackage</code>	3–5
<code>\phi</code>	749, 1315	<code>\restore@mathversion</code>	253
<code>\Pi</code>	716	<code>\Rho</code>	717
		<code>\rho</code>	744, 1314

<code>\rightarrow</code>	1375	<code>\Theta</code>	708
<code>\rootbox</code>	1224	<code>\theta</code>	735
<code>\rppolint</code>	547	<code>\theum@fam</code>	399
S			
<code>\sb</code>	1282	<code>\thinmuskip</code>	1165
<code>\scan_stop:</code>	469, 470, 851, 1245, 1278	<code>\tl_map_inline:nn</code>	236
<code>\scantokens</code>	1145, 1246, 1279	<code>\tl_put_right:NV</code>	1369
<code>\scpolint</code>	548	<code>\tl_remove_in:Nn</code>	8, 285
<code>\scriptscriptstyle</code>	294	<code>\tl_rescan:nn</code>	1150
<code>\scriptstyle</code>	291	<code>\tl_set:Nn</code>	200–202, 204–206, 218–220, 222–224, 358–361, 391, 1248, 1250, 1281, 1283
<code>\set@mathdelimiter</code>	279	<code>\tl_set:Nx</code>	399, 1332, 1334
<code>\set@mathaccent</code>	270	<code>\tl_set_eq:NN</code>	1333
<code>\set@mathchar</code>	272	<code>\to</code>	1375
<code>\set@mathdelimiter</code>	278	<code>\token_if_eq_catcode_p:NN</code>	1343
<code>\set@mathsymbol</code>	273	<code>\token_if_eq_meaning_p:NN</code>	1344
<code>\setbox</code>	24, 32, 71, 79, 1218	<code>\ttdefault</code>	845
<code>\setkeys</code>	362	U	
<code>\setmainfont</code>	5, 40	<code>\um@addto@mathmap</code>	526, 533, <u>538</u>
<code>\SetMathAlphabet</code>	267, 845	<code>\um@backslash</code>	600, 623
<code>\SetMathAlphabet@</code>	268	<code>\um@char@num@range</code> 353, 530, 612, 613, 615	
<code>\setmathfont</code>	20, 67, <u>350</u> , 574	<code>\um@char@range</code> 352, 390, 473, 590, 593, 596	
<code>\SetSymbolFont</code>	262	<code>\um@firstchar</code>	599, 624
<code>\SetSymbolFont@</code>	263	<code>\um@firsttof</code>	622–624
<code>\sf@size</code>	367, 371	<code>\um@font</code>	292, 295, 380, 381, 838, 851, 1219, 1221, 1222, 1225
<code>\shipout</code>	33, 80	<code>\um@fontdimen@percent</code>	287, 292, 295, 1221, 1222
<code>\Sigma</code>	719	<code>\um@mathsymbol</code>	<u>303</u> , 419
<code>\sigma</code>	746	<code>\um@mversion</code>	356, 357
<code>\sp</code>	1249	<code>\um@nolimits</code>	317, <u>544</u> , 552, 560
<code>\space</code>	837, 860	<code>\um@parse@range</code>	609, <u>625</u>
<code>\special</code>	32, 79	<code>\um@parse@term</code>	422, 454, <u>595</u>
<code>\sqint</code>	548	<code>\um@radicals</code>	323, <u>563</u>
<code>\sqrt</code>	563, 1216	<code>\um@resolve@greek</code>	<u>699</u>
<code>\sqrtsign</code>	1216, 1218	<code>\um@scaled@apply</code>	<u>290</u> , 1219, 1225
<code>\std@equal</code>	10	<code>\um@scanactivedef</code>	310, <u>1141</u>
<code>\std@minus</code>	9	<code>\um@scancharlet</code>	<u>1141</u> , 1160
<code>\stepcounter</code>	398	<code>\um@set@mathsymbol</code>	304, <u>305</u>
<code>\string</code>	310, 313, 315, 316, 623, 624	<code>\um@usv@bbLatin</code>	39, 917
<code>\strip@pt</code>	288	<code>\um@usv@bblatin</code>	40, 925
<code>\sumint</code>	546	<code>\um@usv@bbnum</code>	38, 916
T			
<code>\Tau</code>	720	<code>\um@usv@bfDigamma</code>	87, 982
<code>\tau</code>	747	<code>\um@usv@bfdigamma</code>	94, 990
<code>\testmath</code>	15, 25–28, 59, 72–75	<code>\um@usv@bfrakLatin</code>	63, 1085
<code>\tf@size</code>	366, 367	<code>\um@usv@bfraklatin</code>	64, 1086
<code>\the</code>	31, 78		

$\backslash\mathrm{um@usv@bfGreek}$	57, 497, 500, 782, 975, 1014, 1051, 1067, 1071
$\backslash\mathrm{um@usv@bfgreek}$	58, 504, 512, 783, 977, 1021, 1052, 1068, 1072
$\backslash\mathrm{um@usv@bfitGreek}$	61, 497, 500, 784, 976, 1017, 1047, 1051
$\backslash\mathrm{um@usv@bfitgreek}$	62, 504, 512, 785, 978, 1030, 1048, 1052
$\backslash\mathrm{um@usv@bfith}$	104, 979, 1011
$\backslash\mathrm{um@usv@bfitLatin}$	59, 487, 489, 780, 972, 1004, 1045, 1049
$\backslash\mathrm{um@usv@bfitlatin}$	60, 492, 494, 781, 974, 1010, 1046, 1050
$\backslash\mathrm{um@usv@bfitNabla}$ 115, 205, 439, 447, 992, 1054
$\backslash\mathrm{um@usv@bfitpartial}$ 121, 223, 443, 449, 993, 1037, 1055
$\backslash\mathrm{um@usv@bfitvarepsilon}$ 106, 505, 513, 994, 1031, 1056
$\backslash\mathrm{um@usv@bfitvarkappa}$ 108, 507, 515, 996, 1033, 1058
$\backslash\mathrm{um@usv@bfitvarphi}$ 109, 508, 516, 997, 1034, 1059
$\backslash\mathrm{um@usv@bfitvarpi}$ 111, 510, 518, 999, 1036, 1061
$\backslash\mathrm{um@usv@bfitvarrho}$ 110, 509, 517, 998, 1035, 1060
$\backslash\mathrm{um@usv@bfitvarTheta}$ 105, 498, 501, 991, 1018, 1053
$\backslash\mathrm{um@usv@bfitvarthetaeta}$ 107, 506, 514, 995, 1032, 1057
$\backslash\mathrm{um@usv@bfLatin}$	54, 487, 489, 778, 971, 1002, 1049, 1065, 1069
$\backslash\mathrm{um@usv@bflatin}$	55, 56, 492, 494, 779, 973, 1007, 1050, 1066, 1070
$\backslash\mathrm{um@usv@bfNabla}$ 114, 201, 438, 447, 981, 1074
$\backslash\mathrm{um@usv@bfnum}$	53, 967, 1044, 1064, 1084, 1089, 1094, 1110, 1126
$\backslash\mathrm{um@usv@bfpartial}$ 120, 219, 442, 449, 983, 1028, 1075
$\backslash\mathrm{um@usv@bfscrLatin}$	65, 1090
$\backslash\mathrm{um@usv@bfscrlatin}$	66, 1091
$\backslash\mathrm{um@usv@bfsgreek}$	71, 1097, 1113
$\backslash\mathrm{um@usv@bfsgreek}$	72, 1098, 1114
$\backslash\mathrm{um@usv@bfsfitGreek}$	75, 1129
$\backslash\mathrm{um@usv@bfsfitgreek}$	76, 1130
$\backslash\mathrm{um@usv@bfsfitLatin}$	73, 1127
$\backslash\mathrm{um@usv@bfsfitlatin}$	74, 1128
$\backslash\mathrm{um@usv@bfsfitNabla}$ 117, 206, 441, 448, 1132
$\backslash\mathrm{um@usv@bfsfitpartial}$ 123, 224, 445, 450, 1133
$\backslash\mathrm{um@usv@bfslatin}$	68, 1095, 1111
$\backslash\mathrm{um@usv@bfslatin}$	69, 70, 1096, 1112
$\backslash\mathrm{um@usv@bfslNabla}$	116, 202, 440, 448
$\backslash\mathrm{um@usv@bfsgnum}$	67
$\backslash\mathrm{um@usv@bfsgpartial}$	122, 220, 444, 450
$\backslash\mathrm{um@usv@bfsguplatin}$	70
$\backslash\mathrm{um@usv@bfuph}$	103, 1008
$\backslash\mathrm{um@usv@bfuplatin}$	56
$\backslash\mathrm{um@usv@bfvarepsilon}$ 88, 505, 513, 984, 1022, 1076
$\backslash\mathrm{um@usv@bfvarkappa}$ 90, 507, 515, 986, 1024, 1078
$\backslash\mathrm{um@usv@bfvarphi}$ 91, 508, 516, 987, 1025, 1079
$\backslash\mathrm{um@usv@bfvarpi}$ 93, 510, 518, 989, 1027, 1081
$\backslash\mathrm{um@usv@bfvarrho}$ 92, 509, 517, 988, 1026, 1080
$\backslash\mathrm{um@usv@bfvarTheta}$ 86, 498, 501, 980, 1015, 1073
$\backslash\mathrm{um@usv@bfvarthetaeta}$ 89, 506, 514, 985, 1023, 1077
$\backslash\mathrm{um@usv@Digamma}$	78, 968, 982
$\backslash\mathrm{um@usv@digamma}$	85, 969, 990
$\backslash\mathrm{um@usv@frakLatin}$	43, 943, 1085
$\backslash\mathrm{um@usv@fraklatin}$	44, 949, 1086
$\backslash\mathrm{um@usv@itGreek}$	36, 774, 805, 808, 887, 903, 976, 1014, 1017, 1047, 1067, 1097, 1113, 1129
$\backslash\mathrm{um@usv@itgreek}$	37, 814, 822, 888, 904, 978, 1021, 1030, 1048, 1068, 1098, 1114, 1130
$\backslash\mathrm{um@usv@ith}$	95, 770, 797, 800, 902, 979, 1008, 1011
$\backslash\mathrm{um@usv@itLatin}$	32, 768, 789, 791, 885, 900, 917, 928, 943, 953, 958, 963, 972, 1002, 1004, 1045, 1065, 1085, 1090, 1095, 1111, 1127

$\backslash\mathrm{um@usv@itlatin}$	33, 769, 796, 799, 886, 901, 925, 937, 949, 954, 959, 964, 974, 1007, 1010, 1046, 1066, 1086, 1091, 1096, 1112, 1128
$\backslash\mathrm{um@usv@itNabla}$	113, 204, 430, 434, 889, 905, 992, 1039, 1054, 1074, 1100, 1116, 1132
$\backslash\mathrm{um@usv@itpartial}$	119, 222, 432, 435, 890, 906, 993, 1028, 1037, 1040, 1055, 1075, 1101, 1117, 1133
$\backslash\mathrm{um@usv@itvarepsilon}$	97, 815, 823, 892, 908, 994, 1022, 1031, 1056, 1076, 1102, 1118, 1134
$\backslash\mathrm{um@usv@itvarkappa}$	99, 817, 825, 894, 910, 996, 1024, 1033, 1058, 1078, 1104, 1120, 1136
$\backslash\mathrm{um@usv@itvarphi}$	100, 818, 826, 895, 911, 997, 1025, 1034, 1059, 1079, 1105, 1121, 1137
$\backslash\mathrm{um@usv@itvarpi}$	102, 820, 828, 897, 913, 999, 1027, 1036, 1061, 1081, 1107, 1123, 1139
$\backslash\mathrm{um@usv@itvarrho}$	101, 819, 827, 896, 912, 998, 1026, 1035, 1060, 1080, 1106, 1122, 1138
$\backslash\mathrm{um@usv@itvarTheta}$	96, 809, 891, 907, 991, 1015, 1018, 1053, 1073, 1099, 1115
$\backslash\mathrm{um@usv@itvartheta}$	98, 816, 824, 893, 909, 995, 1023, 1032, 1057, 1077, 1103, 1119, 1135
$\backslash\mathrm{um@usv@Nabla}$	112, 200, 429, 434, 889, 905, 981, 1039, 1054, 1074, 1100, 1116, 1132
$\backslash\mathrm{um@usv@num}$	29, 474, 916, 952, 957, 962, 967, 1044, 1064, 1084, 1089, 1094, 1110, 1126
$\backslash\mathrm{um@usv@partial}$	118, 218, 431, 435, 890, 906, 983, 1028, 1037, 1040, 1055, 1075, 1101, 1117, 1133
$\backslash\mathrm{um@usv@scrLatin}$	41, 928
$\backslash\mathrm{um@usv@scrlatin}$	42, 937
$\backslash\mathrm{um@usv@sfitLatin}$	48, 958
$\backslash\mathrm{um@usv@sfitlatin}$	49, 959
$\backslash\mathrm{um@usv@sflatin}$	46, 953
$\backslash\mathrm{um@usv@sflatin}$	47, 954
$\backslash\mathrm{um@usv@sfnum}$	45, 952, 957
$\backslash\mathrm{um@usv@ttLatin}$	51, 963
$\backslash\mathrm{um@usv@ttlLatin}$	52, 964
$\backslash\mathrm{um@usv@ttnum}$	50, 962
$\backslash\mathrm{um@usv@upGreek}$	34, 772, 805, 808, 887, 903, 975, 1014, 1017, 1047, 1067, 1097, 1113, 1129
$\backslash\mathrm{um@usv@upgreek}$	35, 775, 814, 822, 888, 904, 977, 1021, 1030, 1048, 1068, 1098, 1114, 1130
$\backslash\mathrm{um@usv@upLatin}$	30, 767, 789, 791, 885, 900, 917, 928, 943, 953, 958, 963, 971, 1002, 1004, 1045, 1065, 1085, 1090, 1095, 1111, 1127
$\backslash\mathrm{um@usv@uplatin}$	31, 771, 796, 799, 886, 901, 925, 937, 949, 954, 959, 964, 973, 1007, 1010, 1046, 1066, 1086, 1091, 1096, 1112, 1128
$\backslash\mathrm{um@usv@varepsilon}$	79, 815, 823, 892, 908, 984, 1022, 1031, 1056, 1076, 1102, 1118, 1134
$\backslash\mathrm{um@usv@varkappa}$	81, 817, 825, 894, 910, 986, 1024, 1033, 1058, 1078, 1104, 1120, 1136
$\backslash\mathrm{um@usv@varphi}$	82, 818, 826, 895, 911, 987, 1025, 1034, 1059, 1079, 1105, 1121, 1137
$\backslash\mathrm{um@usv@varpi}$	84, 820, 828, 897, 913, 989, 1027, 1036, 1061, 1081, 1107, 1123, 1139
$\backslash\mathrm{um@usv@varrho}$	83, 819, 827, 896, 912, 988, 1026, 1035, 1060, 1080, 1106, 1122, 1138
$\backslash\mathrm{um@usv@varTheta}$	77, 773, 806, 809, 891, 907, 980, 1015, 1018, 1053, 1073, 1099, 1115, 1131
$\backslash\mathrm{um@usv@vartheta}$	80, 816, 824, 893, 909, 985, 1023, 1032, 1057, 1077, 1103, 1119, 1135
$\backslash\mathrm{um@zf@feature}$	<u>566</u> , 578, 581
$\backslash\mathrm{um_bfNabla_up_or_it_usv}$	201, 205, 447, 1039
$\backslash\mathrm{um_bfpartial_up_or_it_usv}$	219, 223, 449, 1040
$\backslash\mathrm{um_bfSfNabla_up_or_it_usv}$	202, 206, 448

<code>\um_bfspartial_up_or_it_usv</code>	<code>\um_peek_execute_branches_ss:</code>
220, 224, 450	1336, 1341
<code>\um_config_mathbb:</code> 915	<code>\um_peek_execute_branches_ss_aux:</code>
<code>\um_config_mathbf:</code> 966	1347, 1360
<code>\um_config_mathbffrak:</code> 1083	<code>\um_prepare_alph:n</code> 834, <u>853</u>
<code>\um_config_mathbfbit:</code> 1043	<code>\um_process_symbol_noparse:nnnn</code>
<code>\um_config_mathbfsc:</code> 1088	393, <u>418</u>
<code>\um_config_mathbfsf:</code> 1093	<code>\um_process_symbol_parse:nnnn</code> 400, <u>418</u>
<code>\um_config_mathbfsfit:</code> 1125	<code>\um_remap_symbol:nnn</code> . 395, 402, 427,
<code>\um_config_mathbfsfup:</code> 1109	429–432, 434, 435, 438–445, 447–450
<code>\um_config_mathbfup:</code> 1063	<code>\um_remap_symbol_noparse:nnn</code> 395, <u>426</u>
<code>\um_config_mathfrak:</code> 942	<code>\um_remap_symbol_parse:nnn</code> 402, <u>426</u> , 453
<code>\um_config_mathit:</code> 899	<code>\um_remap_symbols:</code> 409, <u>426</u>
<code>\um_config_mathscr:</code> 927	<code>\um_scan_sscript:</code> 1251, 1284, 1321, 1323
<code>\um_config_mathsf:</code> 951	<code>\um_scan_sscript:TF</code> 1322, 1331
<code>\um_config_mathsfrit:</code> 956	<code>\um_scanprime:</code>
<code>\um_config_mathhtt:</code> 961	1187, 1197, 1209, 1213, 1214
<code>\um_config_mathup:</code> 884	<code>\um_scanprime_collect:</code>
<code>\um_fix_mathhtt:</code> 844	1190, 1192, 1195, 1198, 1201
<code>\um_glyph_if_exist:n</code> 850	<code>\um_set_mathalph_range:Nnn</code> <u>681</u>
<code>\um_glyph_if_exist:nTF</code>	<code>\um_set_mathalph_range:nNnn</code>
832, <u>850</u> , 1175, 1178, 1181	681, 691, 694, 697
<code>\um_init_alphabet:n</code> 396, 847	<code>\um_set_mathalphabet_char:Nnn</code>
<code>\um_make_mathactive:nNN</code> 464, <u>466</u>	675, 889–897,
<code>\um_map_char:nn</code> 498, 501,	902, 905–913, 918–924, 929–936,
505–510, 513–518, 671, 770, 773,	938–940, 944–948, 968, 969,
797, 800, 806, 809, 815–820, 823–828	979–999, 1008, 1011, 1015, 1018,
<code>\um_map_char:nn_</code> <u>652</u>	1022–1028, 1031–1037, 1039,
<code>\um_map_chars_greek:nn</code>	1040, 1053–1061, 1073–1081,
. . . 497, 500, 504, 512, 665, 772,	1099–1107, 1115–1123, 1131–1139
774, 775, 782–785, 805, 808, 814, 822	<code>\um_set_mathalphabet_char:Nnnn</code> . . <u>674</u>
<code>\um_map_chars_latin:nn</code>	<code>\um_set_mathalphabet_greek:Nnn</code>
487, 489, 492, 494, 662, 767–769,	696, 887, 888,
771, 778–781, 789, 791, 796, 799	903, 904, 975–978, 1014, 1017,
<code>\um_map_chars_numbers:nn</code> 474, 668	1021, 1030, 1047, 1048, 1051,
<code>\um_map_chars_range:nnn</code>	1052, 1067, 1068, 1071, 1072,
652, 663, 666, 669, 672	1097, 1098, 1113, 1114, 1129, 1130
<code>\um_mathmap:Nnn</code> 394, 401, 677, 684	<code>\um_set_mathalphabet_latin:Nnn</code>
<code>\um_mathmap_noparse:Nnn</code> 394, <u>524</u>	693, 885, 886, 900, 901, 917,
<code>\um_mathmap_parse:Nnn</code> 401, <u>529</u>	925, 928, 937, 943, 949, 953, 954,
<code>\um_maybe_init_alphabet:n</code> 396, 403, 833	958, 959, 963, 964, 971–974, 1002,
<code>\um_Nabla_up_or_it_usv</code> . . 200, 204, 434	1004, 1007, 1010, 1045, 1046,
<code>\um_nprimes:n</code> 1167, 1175, 1178, 1181, 1184	1049, 1050, 1065, 1066, 1069,
<code>\um_nprimes_select:n</code> 1171, 1203	1070, 1085, 1086, 1090, 1091,
<code>\um_partial_up_or_it_usv</code> . 218, 222, 435	1095, 1096, 1111, 1112, 1127, 1128
	<code>\um_set_mathalphabet_numbers:Nnn</code>

690, 916, 952, 957, 962, 967, 1044, 1064, 1084, 1089, 1094, 1110, 1126	
\um_set_mathcode:nnnn 347, 460, 541, 655	
\um_setup_active_subscript:nn 1275, 1289–1316	
\um_setup_active_superscript:nn 1242, 1256–1272	
\um_setup_alphabets: 412, 866	
\um_setup_alphanum: 411, 472	
\um_setup_bf_literals: 484, 777	
\um_setup_Greek: 480, 803	
\um_setup_greek: 481, 812	
\um_setup_Latin: 478, 787	
\um_setup_latin: 479, 794	
\um_setup_literals: 476, 766	
\um_setup_math_alphabet:n 831, 867–882	
\um_setup_mathactives: 410, 463	
\um_setup_mathup: 1374	
\um_setup_nabla: 198, 415	
\um_setup_partial: 216, 416	
\um_setup_shapes: 408, 414	
\um_sub_or_super:n ... 1249, 1282, 1325	
\um_symfont_tl 391, 399, 405, 419, 460, 468, 526, 533, 657	
\UnicodeMathSymbol 393, 400, 1159	
\unless 597	
\updefault 406, 845	
\upGreek 9, 44	
\upgreek 10, 45	
\upint 549	
\upLatin 7, 42	
\uplatin 8, 43	
\Upsilon 721	
\upsilon 748	
\use:c 835, 840, 856, 862	
\use_none:n 403	
\usepackage 3, 4, 6, 38, 39, 41	
	V
	\varepsilon 755
	\varkappa 759
	\varointclockwise 546
	\varphi 760
	\varpi 764
	\varrho 763
	\varsigma 745
	\varTheta 718
	\vartheta 758
	\vbox 24, 32, 71, 79
	\version@elt 250
	\version@list 249
	\voffset 22, 69
	W
	\wd 31, 78
	X
	\xdef 560, 593
	\XeTeXdelcode 329, 334
	\XeTeXdelimiter 328, 333
	\XeTeXmathaccent 338
	\XeTeXmathchardef 312, 467
	\XeTeXmathcode 330, 335, 340, 348
	\XeTeXmathcodenum 470, 1245, 1278
	\XeTeXradical 325
	\Xi 714
	\xi 741
	\XKV@rm 376
	Z
	\Z 924, 948
	\z@ 1218, 1221, 1222, 1226
	\Zeta 706
	\zeta 733
	\zf@family 406
	\zf@fontspec 363
	\zf@update@ff 579, 582