

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	1	5.2	Overcoming <code>\@on-</code> <code>lypreamble</code>	15
2	Specification	1	5.3	Other things	16
2.1	Using multiple fonts	2	6	Fundamentals	17
2.2	Script and scriptscript fonts/features	2	6.1	Enlarging the number of maths families	17
3	Maths input	2	6.2	<code>\DeclareMathSymbol</code> for unicode ranges	17
3.1	Miscellanea	3	6.3	The main <code>\setmathfont</code> macro	19
4	Package options	4	6.4	(Big) operators	27
4.1	Math ‘style’	4	6.5	Radicals	30
4.2	Bold switching	5	6.6	Delimiters	31
4.3	Symbols requiring spe- cial attention	6	6.7	Maths accents	33
I	The unicode-math pack- age	7	7	Font features	34
5	Things we need	9	7.1	OpenType maths font features	35
5.1	Package options	12	7.2	Script and scriptscript font options	35
			7.3	Range processing	35

8 Maths alphabets mapping definitions	43	the NFSS	63
8.1 Bold alphabets' character mappings	48	A.1 Overview	63
8.2 Definitions of the math symbols	54	III X_YTeX math font dimensions	64
9 Epilogue	54	IV Some manner of unit testing	69
II stix table data extraction	62	B The regular weight alphabets	70
A Documenting maths support in		C The bold alphabets	71

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 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

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

Figure 1: The unicode superscripts.

0	1	1	2	3	4	5	6	7	8	9	+	-	=	()	□	□	□	i	□	r	u	v	□	β	γ	ε	φ	χ
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Figure 2: The unicode subscripts.

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`, `\primetriples`, 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 ?? and ?. 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 LET-
 TER 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, \TeX 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 \LaTeX ’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 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 8 shows every character under the effect of this package option.

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)	$(\alpha, \beta, \Gamma, \Xi)$

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{f}` in the former (' \mathbf{M} '), and `\bm` (or `\boldsymbol`, deprecated) in the latter (' $\boldsymbol{\xi}$ ').

In `unicode-math`, the `\mathbf{f}` 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 8 shows every character under the effect of this package option.

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). $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ classically uses an upright

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	\mathbb{N}
Italic	Serif	∇
	Bold serif	$\boldsymbol{\nabla}$
	Bold sans	\mathbb{N}

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 argues otherwise) `partial=italic`.¹

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

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.

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	∂

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

(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
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΘΣΤΥΦΧΨΩ
αβγδεζηθικλμνξοπρςστυφχψωεθκφρω

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΘΣΤΥΦΧΨΩ
αβγδεζηθικλμνξοπρςστυφχψωεθκφρω

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΘΣΤΥΦΧΨΩ
αβγδεζηθικλμνξοπρςστυφχψωεθκφρω

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

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

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

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@itLatin{"1D434}
33 \def\um@usv@itlatin{"1D44E}
34 \def\um@usv@upGreek{"391}
35 \def\um@usv@upgreek{"3B1}
36 \def\um@usv@itGreek{"1D6E2}
37 \def\um@usv@itgreek{"1D6FC}
38 \def\um@usv@bnum{"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@sfnun{"1D7E2}
46 \def\um@usv@sflatin{"1D5A0}
47 \def\um@usv@sflatin{"1D5BA}
48 \def\um@usv@sfitLatin{"1D608}
49 \def\um@usv@sfitlatin{"1D622}
50 \def\um@usv@ttnum{"1D7F6}
51 \def\um@usv@ttlLatin{"1D670}
52 \def\um@usv@ttlLatin{"1D68A}

```

Bold:

```

53 \def\um@usv@bfnum{"1D7CE}
54 \def\um@usv@bflatin{"1D400}
55 \def\um@usv@bflatin{"1D41A}
56 \let\um@usv@bfuplatin\um@usv@bflatin
57 \def\um@usv@bfgreek{"1D6A8}
58 \def\um@usv@bfgreek{"1D6C2}
59 \def\um@usv@bfitLatin{"1D468}

```

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

```

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

```

Greek variants:

```

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

```

Bold:

```

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

```

Italic Greek variants:

```

95 \def\um@usv@ith{"210E}
96 \def\um@usv@itvarTheta{"1D6F3}
97 \def\um@usv@itvarepsilon{"1D716}
98 \def\um@usv@itvartheta{"1D717}
99 \def\um@usv@itvarkappa{"1D718}
100 \def\um@usv@itvarphi{"1D719}
101 \def\um@usv@itvarrho{"1D71A}

```

```
102 \def\um@usv@itvarpi{"1D71B}
```

Bold:

```
103 \def\um@usv@bfuph{"1D421}
104 \def\um@usv@bfith{"1D489}
105 \def\um@usv@bfivarTheta{"1D72D}
106 \def\um@usv@bfivarepsilon{"1D750}
107 \def\um@usv@bfivartheta{"1D751}
108 \def\um@usv@bfivarkappa{"1D752}
109 \def\um@usv@bfivarphi{"1D753}
110 \def\um@usv@bfivarrho{"1D754}
111 \def\um@usv@bfivarpi{"1D755}
```

Nabla:

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

Partial:

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

5.1 Package options

xkeyval's package support is used here.

math-style

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

```

137     \bool_set_false:N \g_um_texgreek_bool
138   \or
139     \@um@upGreektrue
140     \@um@upgreekfalse
141     \@um@upLatinfalse
142     \@um@uplatinfalse
143     \@um@b fupGreektrue
144     \@um@b fupgreekfalse
145     \@um@uppartialfalse
146     \@um@b fupLatintrue
147     \@um@b fuplatintrue
148     \@um@upNablatrue
149     \bool_set_true:N \g_um_texgreek_bool
150   \or
151     \@um@upGreektrue
152     \@um@upgreektrue
153     \@um@upLatintrue
154     \@um@uplatinfalse
155     \@um@b fupGreektrue
156     \@um@b fupgreektrue
157     \@um@uppartialtrue
158     \@um@b fupLatintrue
159     \@um@b fuplatintrue
160     \@um@upNablatrue
161     \bool_set_false:N \g_um_texgreek_bool
162   \or
163     \@um@literaltrue
164     \@um@b fliteraltrue
165     \bool_set_false:N \g_um_texgreek_bool
166   \fi
167 }

```

bold-style

```

168 \define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,french,literal}{
169   \ifcase\@tempb\relax
170     \@um@b fupGreekfalse
171     \@um@b fupgreekfalse
172     \@um@uppartialfalse
173     \@um@b fupLatinfalse
174     \@um@b fuplatinfalse
175   \or
176     \@um@b fupGreektrue
177     \@um@b fupgreekfalse
178     \@um@uppartialfalse
179     \@um@b fupLatintrue
180     \@um@b fuplatintrue

```

```

181 \or
182 \um@b fupGreektrue
183 \um@b fupgreektrue
184 \um@uppartialtrue
185 \um@b fupLatintrue
186 \um@b fuplatintrue
187 \or
188 \um@b fliteraltrue
189 \fi
190 }

```

Symbol obliqueness

```

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

```

```

225 \fi
226 }

```

Epsilon and phi shapes

```

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

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

```

5.2 Overcoming \@onlypreamble

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

```

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

```



```

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

```

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>

```

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

```

`\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.

```

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

```

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

```

301 \def\new@mathgroup{\alloc@8\mathgroup\chardef\@cclvi}
302 \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
303 \def \um@mathsymbol#1#2#3#4{
304   \expandafter\um@set@mathsymbol\csname sym#3\endcsname#1#2{#4}}

```

The final macros that actually define the maths symbol with X_YTeX 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.

```

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

```

Operators In the examples following, say we're defining for the symbol $\sum(\Sigma)$.

```
306 \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 $\sum@op$.

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

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

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

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

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

```
322 \else
```

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

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

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

```
327 \ifx\mathopen#3\relax
328 \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
329 \global\XeTeXdelcode#4=#1 #4\relax
```

```

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

```

Accents

```

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

```

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

```

`\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.

```

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

```

6.3 The main `\setmathfont` macro

Here's the simplest usage:

$Ax \triangleq \nabla \times \mathcal{L}$	<pre> \setmathfont{Asana Math} \$Ax \eqdef \nabla \times \mathscr{L} </pre>
---	---

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

$F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st} f(t) dt$	<pre> \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} \[F(s)=\mscrL{\{f(t)\}}=\int_0^\infty \mathup{e}^{-st}f(t)\,,\mathup{d} t \] </pre>
--	---

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

```
350 \DeclareDocumentCommand \setmathfont { O{ } m } {
```

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

```
351 \let\glb@currsizel\relax
```

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

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

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

- Tell fontspec that maths font features are actually allowed.

```
354 \@um@fontspec@featuretrue
```

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

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

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

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

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

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

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

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

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

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

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

```
362 \setkeys*[um]{options}{#1}
```

```
363 \edef\@tempa{\noexpand\zf@fontspec{
```

```

364     Script = Math,
365     SizeFeatures = {
366       {Size = \tf@size-} ,
367       {Size = \sf@size-\tf@size ,
368         Font = \l_um_script_font_tl ,
369         \l_um_script_features_tl
370       } ,
371       {Size = -\sf@size ,
372         Font = \l_um_sscript_font_tl ,
373         \l_um_sscript_features_tl
374       }
375     },
376     \XKV@rm
377   }{#2}
378 }
379 \@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`:

```

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

```

390 \ifx\um@char@range\@empty
391   \tl_set:Nn \um_symfont_tl {um@allsym}
392   \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
393   \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
394   \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
395   \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
396   \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
397 \else
398   \stepcounter{um@fam}
399   \tl_set:Nx \um_symfont_tl {um@fam\theum@fam}
400   \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn

```

```

401 \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
402 \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
403 \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
404 \fi

```

Now defined `\um_symfont_t1` as the \LaTeX math font to access everything:

```

405 \DeclareSymbolFont{\um_symfont_t1}
406 {\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`.

```

407 \@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.)

```

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

```

End of the `\setmathfont` macro.

```

413 }

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

```

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.

```

418 \cs_set:Nn \um_process_symbol_noparse:nnnn {
419   \um@mathsymbol{#2}{#3}{\um_symfont_t1}{#1}
420 }

```

```

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

```

`\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
426 \cs_new:Nn \um_remap_symbols: {
427   \um_remap_symbol: nnn{"2D}{\mathbin}{\hyphen}
428   \if@um@literal
429     \um_remap_symbol: nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
430     \um_remap_symbol: nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
431     \um_remap_symbol: nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
432     \um_remap_symbol: nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
433   \else
434     \um_remap_symbol: nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um@usv@Nabla_up_or_it_usv}
435     \um_remap_symbol: nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um@usv@partial_up_or_it_usv}
436   \fi

```

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

```

437   \if@um@bfliteral
438     \um_remap_symbol: nnn {\um@usv@bNabla}{\mathord}{\um@usv@bNabla}
439     \um_remap_symbol: nnn {\um@usv@bfitNabla}{\mathord}{\um@usv@bfitNabla}
440     \um_remap_symbol: nnn {\um@usv@bfsNabla}{\mathord}{\um@usv@bfsNabla}
441     \um_remap_symbol: nnn {\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsfitNabla}
442     \um_remap_symbol: nnn {\um@usv@bfpartial}{\mathord}{\um@usv@bfpartial}
443     \um_remap_symbol: nnn {\um@usv@bfitpartial}{\mathord}{\um@usv@bfitpartial}
444     \um_remap_symbol: nnn {\um@usv@bfsfpartial}{\mathord}{\um@usv@bfsfpartial}
445     \um_remap_symbol: nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
446   \else
447     \um_remap_symbol: nnn {\um@usv@bNabla,\um@usv@bfitNabla}{\mathord}{\um@usv@bNabla_up_or_it_usv}
448     \um_remap_symbol: nnn {\um@usv@bfsNabla,\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsNabla_up_or_it_usv}
449     \um_remap_symbol: nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um@usv@bfpartial_up_or_it_usv}
450     \um_remap_symbol: nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfpartial_up_or_it_usv}
451   \fi
452 }

```

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

```

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

```



```

460 \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_t1} {#3}
461 }
462 }

```

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:`

```

463 \cs_new:Nn \um_setup_mathactives: {
464   \um_make_mathactive:nnn {"2032} \primesingle \mathord
465   \um_make_mathactive:nnn {"2080} \subscriptzero \mathalpha
466   \um_make_mathactive:nnn {"2081} \subscriptone \mathalpha
467   \um_make_mathactive:nnn {"2082} \subscripttwo \mathalpha
468 }

```

`\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!

```

469 \cs_new:Nn \um_make_mathactive:nnn {
470   \XeTeXmathchardef #2 = "\mathchar@type #3
471                               \csname sym\um_symfont_t1\endcsname
472                               #1 \scan_stop:
473   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:
474 }

```

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`.

```

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

```

Normal weight

```

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

```

Bold

```
486 \if@um@bfliteral
487 \um_setup_bf_literals:
488 \else
489 \if@um@bfupLatin
490 \um_map_chars_latin: nn {\um@usv@bflatin, \um@usv@bfitlatin}{\um@usv@bflatin}
491 \else
492 \um_map_chars_latin: nn {\um@usv@bflatin, \um@usv@bfitlatin}{\um@usv@bfitlatin}
493 \fi
494 \if@um@bfuplatin
495 \um_map_chars_latin: nn {\um@usv@bflatin, \um@usv@bfitlatin}{\um@usv@bflatin}
496 \else
497 \um_map_chars_latin: nn {\um@usv@bflatin, \um@usv@bfitlatin}{\um@usv@bfitlatin}
498 \fi
499 \if@um@bfupgreek
500 \um_map_chars_greek: nn {\um@usv@bfgreek, \um@usv@bfitgreek}{\um@usv@bfgreek}
501 \um_map_char: nn {\um@usv@bfvartheta, \um@usv@bfitvartheta}{\um@usv@bfvartheta}
502 \else
503 \um_map_chars_greek: nn {\um@usv@bfgreek, \um@usv@bfitgreek}{\um@usv@bfitgreek}
504 \um_map_char: nn {\um@usv@bfvartheta, \um@usv@bfitvartheta}{\um@usv@bfitvartheta}
505 \fi
506 \if@um@bfupgreek
507 \um_map_chars_greek: nn {\um@usv@bfgreek, \um@usv@bfitgreek}{\um@usv@bfgreek}
508 \um_map_char: nn {\um@usv@bfvarepsilon, \um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
509 \um_map_char: nn {\um@usv@bfvartheta, \um@usv@bfitvartheta}{\um@usv@bfvartheta}
510 \um_map_char: nn {\um@usv@bfvarkappa, \um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
511 \um_map_char: nn {\um@usv@bfvarphi, \um@usv@bfitvarphi}{\um@usv@bfvarphi}
512 \um_map_char: nn {\um@usv@bfvarrho, \um@usv@bfitvarrho}{\um@usv@bfvarrho}
513 \um_map_char: nn {\um@usv@bfvarpi, \um@usv@bfitvarpi}{\um@usv@bfvarpi}
514 \else
515 \um_map_chars_greek: nn {\um@usv@bfgreek, \um@usv@bfitgreek}{\um@usv@bfitgreek}
516 \um_map_char: nn {\um@usv@bfvarepsilon, \um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
517 \um_map_char: nn {\um@usv@bfvartheta, \um@usv@bfitvartheta}{\um@usv@bfitvartheta}
518 \um_map_char: nn {\um@usv@bfvarkappa, \um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
519 \um_map_char: nn {\um@usv@bfvarphi, \um@usv@bfitvarphi}{\um@usv@bfitvarphi}
520 \um_map_char: nn {\um@usv@bfvarrho, \um@usv@bfitvarrho}{\um@usv@bfitvarrho}
521 \um_map_char: nn {\um@usv@bfvarpi, \um@usv@bfitvarpi}{\um@usv@bfitvarpi}
522 \fi
523 \fi
524 \else
: TODO : what is supposed to happen here?
525 \fi
526 }
```

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.

```

527 \cs_set:Nn \um_mathmap_noparse:Nnn {
528   \clist_map_inline:nn {#2} {
529     \exp_args:No \um@addto@mathmap \um_symfont_tl {##1}{#1}{#3}
530   }
531 }
```

`\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`).

```

532 \cs_set:Nn \um_mathmap_parse:Nnn {
533   \clist_map_inline:Nn \um@char@num@range {
534     \ifnum##1=#3\relax
535       \clist_map_inline:nn {#2} {
536         \exp_args:No \um@addto@mathmap \um_symfont_tl {####1}{#1}{#3}
537       }
538     \fi
539   }
540 }
```

`\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.

```

541 \newcommand\um@addto@mathmap[4]{
542   \expandafter\g@addto@macro
543     \csname um_setup_\cs_to_str:N #3:\endcsname{
544     \um_set_mathcode: nnnn{#2}{\mathalpha}{#1}{#4}
545   }
546 }
```

6.4 (Big) operators

Turns out that \XeTeX 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_{0}^1	<code>\Bbbsum</code>	DOUBLE-STRUCK N-ARY SUMMATION
U+0220F	\prod_{0}^1	<code>\prod</code>	PRODUCT OPERATOR
U+02210	\coprod_{0}^1	<code>\coprod</code>	COPRODUCT OPERATOR
U+02211	\sum_{0}^1	<code>\sum</code>	SUMMATION OPERATOR
U+0222B	\int_0^1	<code>\int</code>	INTEGRAL OPERATOR
U+0222C	\iint_0^1	<code>\iint</code>	DOUBLE INTEGRAL OPERATOR
U+0222D	\iiint_0^1	<code>\iiint</code>	TRIPLE INTEGRAL OPERATOR
U+0222E	\oint_0^1	<code>\oint</code>	CONTOUR INTEGRAL OPERATOR
U+0222F	\oiint_0^1	<code>\oiint</code>	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230	\oiint_0^1	<code>\oiint</code>	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231	\int_0^1	<code>\intclockwise</code>	CLOCKWISE INTEGRAL
U+02232	\oint_0^1	<code>\varointclockwise</code>	CONTOUR INTEGRAL, CLOCKWISE
U+02233	\oint_0^1	<code>\ointctrackwise</code>	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0	\bigwedge_{0}^1	<code>\bigwedge</code>	LOGICAL OR OPERATOR
U+022C1	\bigvee_{0}^1	<code>\bigvee</code>	LOGICAL AND OPERATOR
U+022C2	\bigcap_{0}^1	<code>\bigcap</code>	INTERSECTION OPERATOR
U+022C3	\bigcup_{0}^1	<code>\bigcup</code>	UNION OPERATOR
U+027D5	$\left\lrcorner_{0}^1$	<code>\leftouterjoin</code>	LEFT OUTER JOIN
U+027D6	$\right\lrcorner_{0}^1$	<code>\rightouterjoin</code>	RIGHT OUTER JOIN

U+027D7	$\overset{1}{\times}$	<code>\fullouterjoin</code>	FULL OUTER JOIN
U+027D8	$\overset{1}{\perp}$	<code>\bigbot</code>	LARGE UP TACK
U+027D9	$\overset{1}{\top}$	<code>\bigtop</code>	LARGE DOWN TACK
U+029F8	$\overset{1}{/}$	<code>\xsol</code>	BIG SOLIDUS
U+029F9	$\overset{1}{\backslash}$	<code>\xbsol</code>	BIG REVERSE SOLIDUS
U+02A00	$\overset{1}{\odot}$	<code>\bigodot</code>	N-ARY CIRCLED DOT OPERATOR
U+02A01	$\overset{1}{\oplus}$	<code>\bigoplus</code>	N-ARY CIRCLED PLUS OPERATOR
U+02A02	$\overset{1}{\otimes}$	<code>\bigotimes</code>	N-ARY CIRCLED TIMES OPERATOR
U+02A03	$\overset{1}{\cup\cdot}$	<code>\bigcupdot</code>	N-ARY UNION OPERATOR WITH DOT
U+02A04	$\overset{1}{\cup+}$	<code>\bigcupplus</code>	N-ARY UNION OPERATOR WITH PLUS
U+02A05	$\overset{1}{\sqcap}$	<code>\bigsqcap</code>	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06	$\overset{1}{\sqcup}$	<code>\bigsqcup</code>	N-ARY SQUARE UNION OPERATOR
U+02A07	$\overset{1}{\&}$	<code>\conjquant</code>	TWO LOGICAL AND OPERATOR
U+02A08	$\overset{1}{\vee}$	<code>\disjquant</code>	TWO LOGICAL OR OPERATOR
U+02A09	$\overset{1}{\times}$	<code>\bigtimes</code>	N-ARY TIMES OPERATOR
U+02A0B	$\overset{1}{\int}_0^1$	<code>\sumint</code>	SUMMATION WITH INTEGRAL
U+02A0C	$\overset{1}{\iiint}_0^1$	<code>\iiiint</code>	QUADRUPLE INTEGRAL OPERATOR
U+02A0D	$\overset{1}{f}_0^1$	<code>\intbar</code>	FINITE PART INTEGRAL
U+02A0E	$\overset{1}{\oint}_0^1$	<code>\intBar</code>	INTEGRAL WITH DOUBLE STROKE
U+02A0F	$\overset{1}{f}_0^1$	<code>\fint</code>	INTEGRAL AVERAGE WITH SLASH
U+02A10	$\overset{1}{f}_0^1$	<code>\cirfnint</code>	CIRCULATION FUNCTION
U+02A11	$\overset{1}{f}_0^1$	<code>\awint</code>	ANTICLOCKWISE INTEGRATION LINE INTEGRATION WITH RECTANGULAR
U+02A12	$\overset{1}{f}_0^1$	<code>\rppoint</code>	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13	$\overset{1}{f}_0^1$	<code>\scpoint</code>	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE
U+02A14	$\overset{1}{f}_0^1$	<code>\npoint</code>	POLE

U+02A15		<code>\pointint</code>	INTEGRAL AROUND A POINT OPERATOR
U+02A16		<code>\sqint</code>	QUATERNION INTEGRAL OPERATOR
U+02A17		<code>\intlarhk</code>	INTEGRAL WITH LEFTWARDS ARROW WITH 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 17). 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.

```

547 \def\um@nolimits{
548   \@elt\int\@elt\iint\@elt\iiint\@elt\iiint\@elt\oint\@elt\oiint\@elt\oiint
549   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
550   \@elt\intbar\@elt\intBar\@elt\oint\@elt\cirfnint\@elt\awint\@elt\rppoint
551   \@elt\scpolint\@elt\ntopolint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
552   \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
553 }

```

`\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`.

```

554 \newcommand\addnolimits[1]{
555   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
556 }

```

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

```

557 \def\removenolimits#1{
558   \begingroup
559     \def\@elt##1{
560       \ifx##1#1\else
561         \noexpand\@elt\noexpand##1
562       \fi}
563     \xdef\um@nolimits{\um@nolimits}
564   \endgroup
565 }

```

$$\iiint_V \iiint_V \iiint_V$$

```

\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 `\cuberoot` 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.

```

566 \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/ltxbugs2html?pr=latex/3853&prlatex/3754>

```
567 \let\left@primitive\left
568 \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`

$$\left(\left(\left((x^1)^2\right)^3\right)^4\right)^5$$






















$$\left[\left[\left[[y^1]^2\right]^3\right]^4\right]^5$$

$$\left\{\left\{\left\{[z^1]^2\right\}^3\right\}^4\right\}^5$$

```
\setmathfont{Cambria Math}
\[ \left(\left(\left(\left(\left(\left( x
\right)^1\right)\right)^2\right)\right)^3\right)\right)^4\right)^5 \]
\[ \left[\left[\left[\left[\left[\left[ y
\right]^1\right]\right]\right]^2\right]\right]^3\right]\right)^4\right]^5 \]
\[ \left\{\left\{\left\{\left\{\left\{\left\{ z
\right\}^1\right\}\right\}^2\right\}\right\}^3\right\}\right\}^4\right\}^5 \]
```

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>\sqr t</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
U+02772	⌞	<code>\lbrbrak</code>	LIGHT LEFT TORTOISE SHELL BRACKET
			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	‘	<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

U+02773		<code>\rbrbrak</code>	LIGHT RIGHT TORTOISE SHELL BRACKET
U+027C6	⌋	<code>\rbag</code>	ORNAMENT
U+027E7	⌋	<code>\rBrack</code>	RIGHT S-SHAPED BAG DELIMITER
U+027E9	⌋	<code>\rangle</code>	MATHEMATICAL RIGHT WHITE SQUARE
U+027EB	⌋	<code>\rAngle</code>	BRACKET
U+027ED		<code>\Rbrbrak</code>	MATHEMATICAL RIGHT DOUBLE ANGLE
U+02984	⌋	<code>\Rbrace</code>	SHELL BRACKET
U+02986	⌋	<code>\rParen</code>	RIGHT WHITE CURLY BRACKET
U+02988	⌋	<code>\rrparenthesis</code>	RIGHT WHITE PARENTHESIS
U+0298A	⌋	<code>\rrangle</code>	Z NOTATION RIGHT IMAGE BRACKET
U+0298C	⌋	<code>\rbrackubar</code>	Z NOTATION RIGHT BINDING BRACKET
U+0298E	⌋	<code>\rbracklrtick</code>	RIGHT SQUARE BRACKET WITH UNDERBAR
U+02990	⌋	<code>\rbrackurtick</code>	RIGHT SQUARE BRACKET WITH TICK IN
U+02992	⌋	<code>\rangledot</code>	CORNER
U+02994	⌋	<code>\rparengtr</code>	RIGHT ANGLE BRACKET WITH DOT
U+02998	⌋	<code>\rblkrbrak</code>	RIGHT ARC GREATER-THAN BRACKET
U+029D9	⌋	<code>\rvzigzag</code>	RIGHT BLACK TORTOISE SHELL BRACKET
U+029DB	⌋	<code>\Rvzigzag</code>	RIGHT WIGGLY FENCE
U+029FD	⌋	<code>\rcurvyangle</code>	RIGHT DOUBLE WIGGLY FENCE
U+03015		<code>\rbrbrak</code>	RIGHT POINTING CURVED ANGLE BRACKET
U+03019		<code>\Rbrbrak</code>	RIGHT BROKEN BRACKET
			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	˘	<code>\grave</code>	GRAVE ACCENT
U+00301	ˆ	<code>\acute</code>	ACUTE ACCENT
U+00302	ˆ	<code>\hat</code>	CIRCUMFLEX ACCENT
U+00303	˜	<code>\tilde</code>	TILDE
U+00304	¯	<code>\bar</code>	MACRON
U+00305	̄	<code>\overbar</code>	OVERBAR EMBELLISHMENT
U+00306	˘	<code>\breve</code>	BREVE
U+00307	˙	<code>\dot</code>	DOT ABOVE
U+00308	¨	<code>\ddot</code>	DIERESIS
U+00309	˘	<code>\ovhook</code>	COMBINING HOOK ABOVE
U+0030A	ˆ	<code>\ocirc</code>	RING

U+0030C	Š	\check	CARON
U+00310	Ṡ	\candra	CANDRABINDU (NON-SPACING)
U+00312	ˆ	\oturnedcomma	COMBINING TURNED COMMA ABOVE
U+00313	˘	\osmooth	GREEK PSILI (SMOOTH BREATHING) (NON-SPACING)
U+00314	͂	\orough	GREEK DASIA (ROUGH BREATHING) (NON-SPACING)
U+00315	̓	\ocommatopright	COMBINING COMMA ABOVE RIGHT
U+0031A	͆	\droang	LEFT ANGLE ABOVE (NON-SPACING)
U+020D0	ͅ	\leftharpoonaccent	COMBINING LEFT HARPOON ABOVE
U+020D1	͇	\rightharpoonaccent	COMBINING RIGHT HARPOON ABOVE
U+020D2	͈	\vertoverlay	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6	͊	\overleftarrow	COMBINING LEFT ARROW ABOVE
U+020D7	͋	\vec	COMBINING RIGHT ARROW ABOVE
U+020DB	͍	\dddot	COMBINING THREE DOTS ABOVE
U+020DC	͎	\ddddot	COMBINING FOUR DOTS ABOVE
U+020E1	͑	\overleftrightharpoon	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7	͗	\annuity	COMBINING ANNUITY SYMBOL
U+020E8	͘	\threeunderdot	COMBINING TRIPLE UNDERDOT
U+020E9	͙	\widebridgeabove	COMBINING WIDE BRIDGE ABOVE
U+020EC	͜	\underrightharpoondown	COMBINING RIGHTWARDS HARPOON WITH BARB DOWNWARDS
U+020ED	͝	\underleftharpoondown	COMBINING LEFTWARDS HARPOON WITH BARB DOWNWARDS
U+020EE	͞	\underleftarrow	COMBINING LEFT ARROW BELOW
U+020EF	͟	\underrightarrow	COMBINING RIGHT ARROW BELOW
U+020F0	͠	\asteraccent	COMBINING ASTERISK ABOVE

7 Font features

`\um@z f@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.

```

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

```

```

579 }
580 }

```

7.1 OpenType maths font features

```

581 \um@zf@feature{ScriptStyle}{
582   \zf@update@ff{+ssty=0}
583 }
584 \um@zf@feature{ScriptScriptStyle}{
585   \zf@update@ff{+ssty=1}
586 }

```

7.2 Script and scriptscript font options

```

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

```

7.3 Range processing

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

```

591 \define@choicekey+[um]{options}{Range}[ \@tempa\@tempb]{ALL}{
592   \ifcase\@tempb\relax
593     \global\let\um@char@range\@empty
594   \fi
595 }{
596   \xdef\um@char@range{#1}
597 }

```

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:

```

598 \newcommand\um@parse@term[4]{
599   \clist_map_variable:NNn \um@char@range \@ii {

```

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

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

```
600 \unless\ifx\@ii\@empty
601 \@tempswa false
```

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

```
602 \expandafter\um@firstchar\expandafter{\@ii}
603 \ifx\@tempa\um@backslash
604 \expandafter\ifx\@ii#2\relax
605 \@tempswa true
606 \else
607 \expandafter\ifx\@ii#3\relax
608 \@tempswa true
609 \fi
610 \fi
```

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

```
611 \else
612 \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
613 \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.

```
614 \if@tempswa
615 \ifx\um@char@num@range\@empty
616 \g@addto@macro\um@char@num@range{#1}
617 \else
618 \g@addto@macro\um@char@num@range{, #1}
619 \fi
620 #4%
621 \fi
622 \fi
623 }
624 }
625 \def\um@firstof#1#2\@nil{#1}
626 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
627 \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

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

```

645         \else
646         \ifnum#4>\numexpr#1-1\relax
647         \ifnum#4<\numexpr#2+1\relax
648         \@tempwattrue
649         \fi
650         \fi
651         \fi
652     \fi
653 \fi
654 }

\um_map_char: nn    #1 : Number of iterations
                   #2 : Starting input char(s)
                   #3 : Starting output char
                   Loops through character ranges setting \mathcode.
655 \cs_set:Nn \um_map_chars_range: nnn {
656     \clist_map_variable: nNn {#2} \l_um_input_num {
657         \prg_stepwise_variable: nnnNn{0}{1}{#1} \l_um_incr_num {
658             \um_set_mathcode: nnnn
659             {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
660             {\mathalpha}{\um_symfont_t1}
661             {\numexpr \l_um_incr_num + #3 \relax}
662         }
663     }
664 }
665 \cs_set:Nn \um_map_chars_latin: nn {
666     \um_map_chars_range: nnn {25}{#1}{#2}
667 }
668 \cs_set:Nn \um_map_chars_greek: nn {
669     \um_map_chars_range: nnn {24}{#1}{#2}
670 }
671 \cs_set:Nn \um_map_chars_numbers: nn {
672     \um_map_chars_range: nnn {9}{#1}{#2}
673 }
674 \cs_set:Nn \um_map_char: nn {
675     \um_map_chars_range: nnn {0}{#1}{#2}
676 }

```

```

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

```

```

681      {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
682    }
683  }

```

`\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`.

```

684 \cs_new:Nn \um_set_mathalph_range:nNnn {
685   \clist_map_variable:nNn {#3} \l_um_input_num {
686     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
687       \exp_args:Nnff \um_mathmap:Nnn {#2}
688       {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
689       {\number\numexpr \l_um_inc_num + #4 \relax}
690     }
691   }
692 }
693 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
694   \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
695 }
696 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
697   \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
698 }
699 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
700   \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
701 }

```

BCDBCDEABCDEF

```

\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.

```

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

```



```

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

```

Lowercase:

```

729 \def\alpha{\mitalpha}
730 \def\beta{\mitbeta}
731 \def\gamma{\mitgamma}
732 \def\delta{\mitdelta}
733 \def\epsilon{
734   \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
735 }
736 \def\zeta{\mitzeta}
737 \def\eta{\miteta}
738 \def\theta{\mittheta}
739 \def\iota{\mitiota}
740 \def\kappa{\mitkappa}
741 \def\lambda{\mitlambda}
742 \def\mu{\mitmu}
743 \def\nu{\mitnu}
744 \def\xi{\mitxi}
745 \def\omicron{\mitomicron}
746 \def\pi{\mitpi}
747 \def\rho{\mitrho}
748 \def\varsigma{\mitvarsigma}
749 \def\sigma{\mitsigma}
750 \def\tau{\mittau}
751 \def\upsilon{\mitupsilon}
752 \def\phi{
753   \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}
754 }
755 \def\chi{\mitchi}

```

```

756 \def\psi{\mitpsi}
757 \def\omega{\mitomega}
758 \def\varepsilon{
759   \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
760 }
761 \def\vartheta{\mitvartheta}
762 \def\varkappa{\mitvarkappa}
763 \def\varphi{
764   \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
765 }
766 \def\varrho{\mitvarrho}
767 \def\varpi{\mitvarpi}
768 }

```

\um_setup_literals: : TODO : other literal symbols

```

769 \cs_set:Nn \um_setup_literals: {
770   \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}
771   \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}
772   \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}
773   \um_map_char:nn {\um@usv@ith}{\um@usv@ith}
774   \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}
775   \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}
776   \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}
777   \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}
778   \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}
779 }

```

\um_setup_bf_literals: TODO: other literal symbols

```

780 \cs_set:Nn \um_setup_bf_literals: {
781   \um_map_chars_latin:nn {\um@usv@bflatin}{\um@usv@bflatin}
782   \um_map_chars_latin:nn {\um@usv@bflatin}{\um@usv@bflatin}
783   \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}
784   \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}
785   \um_map_chars_greek:nn {\um@usv@bfgreek}{\um@usv@bfgreek}
786   \um_map_chars_greek:nn {\um@usv@bfgreek}{\um@usv@bfgreek}
787   \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
788   \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
789 }

```

\um_setup_Latin:

```

790 \cs_set:Nn \um_setup_Latin: {
791   \if@um@upLatin
792     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
793   \else
794     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
795   \fi
796 }

```

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

```

797 \cs_set:Nn \um_setup_latin: {
798   \if@um@uplatin
799     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
800     \um_map_char:nn {\um@usv@ith}{`\h}
801   \else
802     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
803     \um_map_char:nn {\`\h,\um@usv@ith}{\um@usv@ith}
804   \fi
805 }

```

`\um_setup_Greek:`

```

806 \cs_set:Nn \um_setup_Greek: {
807   \if@um@upgreek
808     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
809     \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
810   \else
811     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
812     \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}
813   \fi
814 }

```

`\um_setup_greek:`

```

815 \cs_set:Nn \um_setup_greek: {
816   \if@um@upgreek
817     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
818     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
819     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
820     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
821     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
822     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
823     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
824   \else
825     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
826     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
827     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
828     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
829     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
830     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
831     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
832   \fi
833 }

```

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.

```

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

850 \cs_set:Nn \um_init_alphabet:n {
851   \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
852 }

```

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

```

853 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
854   \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
855 }

```

`\um_prepare_alph:n` If `\mathXY` hasn't been (re-)declared yet, then define it in terms of unicode-math definitions.

```

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

```

```

859     \begingroup \use:c {um_setup_math#1:} ##1 \endgroup
860   }
861   \cs_set_protected:cpn {math#1} {
862     \mode_if_math:F {
863       \expandafter\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 {sfit} }
877   \um_setup_math_alphabet:n {tt}   }
878   \um_setup_math_alphabet:n {bf}   }
879   \um_setup_math_alphabet:n {bfup} }
880   \um_setup_math_alphabet:n {bfit} }
881   \um_setup_math_alphabet:n {bfscr} }
882   \um_setup_math_alphabet:n {bffrak} }
883   \um_setup_math_alphabet:n {bfsf} }
884   \um_setup_math_alphabet:n {bfsfup} }
885   \um_setup_math_alphabet:n {bfsfit} }
886 }

```

: TODO : nested alphabets?

8.0.1 Upright: `\mathup`

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

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

```

887 \cs_new:Npn \um_config_mathup: {
888   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
889   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
890   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
891   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}

```

```

892 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@Nabla, \um@usv@itNabla}{\um@usv@Nabla}
893 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@partial, \um@usv@itpartial}{\um@usv@partial}
894 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@varTheta, \um@usv@itvarTheta}{\um@usv@varThe
895 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@varepsilon, \um@usv@itvarepsilon}{\um@usv@va
896 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@vartheta, \um@usv@itvartheta}{\um@usv@varthe
897 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@varkappa, \um@usv@itvarkappa}{\um@usv@varkap
898 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@varphi, \um@usv@itvarphi}{\um@usv@varphi}
899 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@varrho, \um@usv@itvarrho}{\um@usv@varrho}
900 \um_set_mathalphabet_char: Nnn{\mathup}{\um@usv@varpi, \um@usv@itvarpi}{\um@usv@varpi}
901 }

```

8.0.2 Italic: \mathit

<i>ABCDEFGHIJKLMNOPQRSTUVWXYZ</i>	<code>\$\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\\</code>
<i>abcdefghijklmnopqrstuvwxyz</i>	<code>\$\mathit{abcdefghijklmnopqrstuvwxyz}\$ \\\</code>
<i>ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ</i>	<code>\$\mathit{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}\$\quad\$\mathit{Θ}\$ \\\</code>
<i>αβγδεζηθικλμνξοπρστυφχψω εθκφρω</i>	<code>\$\mathit{αβγδεζηθικλμνξοπρστυφχψω}\$\quad\$\mathit{εθκφρω}\$ \\\</code>

Roman:

```

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

```

Greek:

```

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

```

8.0.3 Blackboard or double-struck: \mathbb

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

Numbers:

```

918 \cs_new:Npn \um_config_mathbb: {
919   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@num}{\um@usv@bnum}

```

Roman uppercase:

```

920 \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
921 \um_set_mathalphabet_char:Nnn{\mathbb}{`\C,"1D60A}{\um@usv@bbLatin}
922 \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D60F}{\um@usv@bbLatin}
923 \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D60F}{\um@usv@bbLatin}
924 \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D617}{\um@usv@bbLatin}
925 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D618}{\um@usv@bbLatin}
926 \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D619}{\um@usv@bbLatin}
927 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D621}{\um@usv@bbLatin}

```

Roman lowercase:

```

928 \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
929 }

```

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

`\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}`
`\mathscr{abcdefghijklmnopqrstuvwxyz}`

```

930 \cs_new:Npn \um_config_mathscr: {
931   \um_set_mathalphabet_latin:Nnn{\mathscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}
932   \um_set_mathalphabet_char:Nnn{\mathscr}{`\B,"1D435}{\um@usv@scrLatin}
933   \um_set_mathalphabet_char:Nnn{\mathscr}{`\E,"1D438}{\um@usv@scrLatin}
934   \um_set_mathalphabet_char:Nnn{\mathscr}{`\F,"1D439}{\um@usv@scrLatin}
935   \um_set_mathalphabet_char:Nnn{\mathscr}{`\H,"1D43B}{\um@usv@scrLatin}
936   \um_set_mathalphabet_char:Nnn{\mathscr}{`\I,"1D43C}{\um@usv@scrLatin}
937   \um_set_mathalphabet_char:Nnn{\mathscr}{`\L,"1D43F}{\um@usv@scrLatin}
938   \um_set_mathalphabet_char:Nnn{\mathscr}{`\M,"1D440}{\um@usv@scrLatin}
939   \um_set_mathalphabet_char:Nnn{\mathscr}{`\R,"1D445}{\um@usv@scrLatin}
940   \um_set_mathalphabet_latin:Nnn{\mathscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}
941   \um_set_mathalphabet_char:Nnn{\mathscr}{`\e,"1D452}{\um@usv@scrLatin}
942   \um_set_mathalphabet_char:Nnn{\mathscr}{`\g,"1D454}{\um@usv@scrLatin}
943   \um_set_mathalphabet_char:Nnn{\mathscr}{`\o,"1D45C}{\um@usv@scrLatin}
944 }

```

8.0.5 Fraktur or fraktur or blackletter: `\mathfrak`

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

`\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}`
`\mathfrak{abcdefghijklmnopqrstuvwxyz}`

Letters, with exceptions $\{\mathbb{C}, \mathbb{S}, \mathbb{Z}, \mathbb{R}, \mathbb{J}\}$:

```

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

```

8.0.6 Sans serif: `\mathsf`

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

```

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

```

8.0.7 Sans serif italic: `\mathsfit`

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

```

959 \cs_new:Npn \um_config_mathsfitt: {
960   \um_set_mathalphabet_numbers:Nnn{\mathsfit}{\um@usv@num}{\um@usv@sfnun}
961   \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitlatin}
962   \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlatin}
963 }

```

8.0.8 Typewriter or monospaced: `\mathtt`

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


```

964 \cs_new:Npn \um_config_mathhtt: {
965   \um_set_mathalphabet_numbers:Nnn{\mathhtt}{\um@usv@num}{\um@usv@ttnum}
966   \um_set_mathalphabet_latin:Nnn{\mathhtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
967   \um_set_mathalphabet_latin:Nnn{\mathhtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlLatin}
968 }

```

8.1 Bold alphabets' character mappings

8.1.1 Bold: \mathbf{f}

0123456789

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ

Θ

αβγδεζηθικλμνξοπρστυφχψω

εθκφρϖ

```

$ \mathbf{f}{0123456789}$ \l
$ \mathbf{f}{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \l
$ \mathbf{f}{abcdefghijklmnopqrstuvwxyz}$ \l
$ \mathbf{f}{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$ \quad$ \mathbf{f}{Θ}$ \l
$ \mathbf{f}{αβγδεζηθικλμνξοπρστυφχψω}$ \quad$ \mathbf{f}{εθκφρϖ}$ \l

```

```

969 \cs_new:Npn \um_config_mathbf: {
970   \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@b fnum}
971   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{ "1D7CA}
972   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{ "1D7CB}
973   \if@um@b fliteral
974   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@b fLatin}
975   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@b fi tLatin}
976   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@b flatin}
977   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@b fi tlatin}
978   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@b fGreek}
979   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@b fi tGreek}
980   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@b fgreek}
981   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@b fi tgreek}
982   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@i th}{\um@usv@b fi th}
983   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@var Theta}{\um@usv@b fvar Theta}
984   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@b fNabla}
985   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@b fDigamma}
986   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@b fpartial}
987   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@b fvarepsilon}
988   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@b fvartheta}
989   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@b fvarkappa}
990   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@b fvarphi}
991   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@b fvarrho}
992   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@b fvarpi}
993   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@b fdigamma}
994   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvar Theta}{\um@usv@b fi tvar Theta}

```

```

995 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
996 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
997 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
998 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
999 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1000 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
1001 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
1002 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
1003 \else
1004 \if@um@b fupLatin
1005 \um_set_mathalphabet_latin: Nnn{\mathbf}{\um@usv@upLatin, \um@usv@itLatin}{\um@usv@bflatLatin}
1006 \else
1007 \um_set_mathalphabet_latin: Nnn{\mathbf}{\um@usv@upLatin, \um@usv@itLatin}{\um@usv@bfitLatin}
1008 \fi
1009 \if@um@b fuplatin
1010 \um_set_mathalphabet_latin: Nnn{\mathbf}{\um@usv@uplatin, \um@usv@itlatin}{\um@usv@bflatlatin}
1011 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
1012 \else
1013 \um_set_mathalphabet_latin: Nnn{\mathbf}{\um@usv@uplatin, \um@usv@itlatin}{\um@usv@bfitlatin}
1014 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1015 \fi
1016 \if@um@b fupGreek
1017 \um_set_mathalphabet_greek: Nnn{\mathbf}{\um@usv@upGreek, \um@usv@itGreek}{\um@usv@bfGreek}
1018 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varTheta, \um@usv@itvarTheta}{\um@usv@bfvarTheta}
1019 \else
1020 \um_set_mathalphabet_greek: Nnn{\mathbf}{\um@usv@upGreek, \um@usv@itGreek}{\um@usv@bfitGreek}
1021 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varTheta, \um@usv@itvarTheta}{\um@usv@bfitvarTheta}
1022 \fi
1023 \if@um@b fupgreek
1024 \um_set_mathalphabet_greek: Nnn{\mathbf}{\um@usv@upgreek, \um@usv@itgreek}{\um@usv@bfGreek}
1025 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varepsilon, \um@usv@itvarepsilon}{\um@usv@bfvarTheta}
1026 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@vartheta, \um@usv@itvartheta}{\um@usv@bfvarTheta}
1027 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varkappa, \um@usv@itvarkappa}{\um@usv@bfvarTheta}
1028 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varphi, \um@usv@itvarphi}{\um@usv@bfvarphi}
1029 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varrho, \um@usv@itvarrho}{\um@usv@bfvarrho}
1030 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varpi, \um@usv@itvarpi}{\um@usv@bfvarpi}
1031 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@partial, \um@usv@itpartial}{\um@usv@bfpartial}
1032 \else
1033 \um_set_mathalphabet_greek: Nnn{\mathbf}{\um@usv@upgreek, \um@usv@itgreek}{\um@usv@bfitgreek}
1034 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varepsilon, \um@usv@itvarepsilon}{\um@usv@bfvarTheta}
1035 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@vartheta, \um@usv@itvartheta}{\um@usv@bfvarTheta}
1036 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varkappa, \um@usv@itvarkappa}{\um@usv@bfvarTheta}
1037 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varphi, \um@usv@itvarphi}{\um@usv@bfvarphi}
1038 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varrho, \um@usv@itvarrho}{\um@usv@bfvarrho}
1039 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@varpi, \um@usv@itvarpi}{\um@usv@bfvarpi}
1040 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@partial, \um@usv@itpartial}{\um@usv@bfitpartial}

```

```

1041 \fi
1042 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@Nabla, \um@usv@itNabla}{\um_bfNabla_up_or_i
1043 \um_set_mathalphabet_char: Nnn{\mathbf}{\um@usv@partial, \um@usv@itpartial}{\um_bfpartial_u
1044 \fi
1045 }

```

8.1.2 Bold Italic: `\mathbf{it}`

0123456789	<code>\$\mathbf{it}{0123456789}\$ \</code>
ABCDEFGHIJKLMNOPQRSTUVWXYZ	<code>\$\mathbf{it}{ABCDEFGHIJKLMNopqrstuvwxyZ}\$ \</code>
abcdefghijklmnopqrstuvwxyz	<code>\$\mathbf{it}{abcdefghijklmnopqrstuvwxyz}\$ \</code>
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	<code>\$\mathbf{it}{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}\$ \quad</code>
αβγδεζηθικλμνξοπρστυφχψω εθκφρϖ	<code>\$\mathbf{it}{αβγδεζηθικλμνξοπρστυφχψω}\$ \quad</code>
	<code>\$\mathbf{it}{εθκφρϖ}\$ \</code>

```

1046 \cs_new:Npn \um_config_mathbf{it: {
1047   \um_set_mathalphabet_numbers: Nnn{\mathbf{it}}{\um@usv@num}{\um@usv@bfnun}
1048   \um_set_mathalphabet_latin: Nnn{\mathbf{it}}{\um@usv@upLatin, \um@usv@itLatin}{\um@usv@bfitLat
1049   \um_set_mathalphabet_latin: Nnn{\mathbf{it}}{\um@usv@uplatin, \um@usv@itlatin}{\um@usv@bfitLat
1050   \um_set_mathalphabet_greek: Nnn{\mathbf{it}}{\um@usv@upGreek, \um@usv@itGreek}{\um@usv@bfitGre
1051   \um_set_mathalphabet_greek: Nnn{\mathbf{it}}{\um@usv@upgreek, \um@usv@itgreek}{\um@usv@bfitgre
1052   \um_set_mathalphabet_latin: Nnn{\mathbf{it}}{\um@usv@bflatin}{\um@usv@bfitLatin}
1053   \um_set_mathalphabet_latin: Nnn{\mathbf{it}}{\um@usv@bflatin}{\um@usv@bfitlatin}
1054   \um_set_mathalphabet_greek: Nnn{\mathbf{it}}{\um@usv@bfgreek}{\um@usv@bfitGreek}
1055   \um_set_mathalphabet_greek: Nnn{\mathbf{it}}{\um@usv@bfgreek}{\um@usv@bfitgreek}
1056   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@varTheta, \um@usv@itvarTheta}{\um@usv@bfit
1057   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@Nabla, \um@usv@itNabla}{\um@usv@bfitNabla}
1058   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@partial, \um@usv@itpartial}{\um@usv@bfitpa
1059   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@varepsilon, \um@usv@itvarepsilon}{\um@usv@
1060   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@vartheta, \um@usv@itvartheta}{\um@usv@bfit
1061   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@varkappa, \um@usv@itvarkappa}{\um@usv@bfit
1062   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@varphi, \um@usv@itvarphi}{\um@usv@bfitvarp
1063   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@varrho, \um@usv@itvarrho}{\um@usv@bfitvarr
1064   \um_set_mathalphabet_char: Nnn{\mathbf{it}}{\um@usv@varpi, \um@usv@itvarpi}{\um@usv@bfitvarpi}
1065 }

```

8.1.3 Bold Italic: `\mathbf{fup}`

0123456789	<code>\$\mathbf{fup}{0123456789}\$ \</code>
ABCDEFGHIJKLMNOPQRSTUVWXYZ	<code>\$\mathbf{fup}{ABCDEFGHIJKLMNopqrstuvwxyZ}\$ \</code>
abcdefghijklmnopqrstuvwxyz	<code>\$\mathbf{fup}{abcdefghijklmnopqrstuvwxyz}\$ \</code>
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	<code>\$\mathbf{fup}{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}\$ \quad</code>
αβγδεζηθικλμνξοπρστυφχψω εθκφρϖ	<code>\$\mathbf{fup}{αβγδεζηθικλμνξοπρστυφχψω}\$ \quad</code>
	<code>\$\mathbf{fup}{εθκφρϖ}\$ \</code>

```

1066 \cs_new:Npn \um_config_mathbfup: {
1067   \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnun}
1068   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin, \um@usv@itLatin}{\um@usv@bflatin}
1069   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin, \um@usv@itlatin}{\um@usv@bflatin}
1070   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek, \um@usv@itGreek}{\um@usv@bfgreek}
1071   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek, \um@usv@itgreek}{\um@usv@bfgreek}
1072   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bflatin}{\um@usv@bflatin}
1073   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bflatin}{\um@usv@bflatin}
1074   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfgreek}{\um@usv@bfgreek}
1075   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfgreek}{\um@usv@bfgreek}
1076   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta, \um@usv@itvarTheta}{\um@usv@bfva}
1077   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla, \um@usv@itNabla}{\um@usv@bfNabla}
1078   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial, \um@usv@itpartial}{\um@usv@bfpart}
1079   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon, \um@usv@itvarepsilon}{\um@usv@}
1080   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta, \um@usv@itvartheta}{\um@usv@bfva}
1081   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa, \um@usv@itvarkappa}{\um@usv@bfva}
1082   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi, \um@usv@itvarphi}{\um@usv@bfvarphi}
1083   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho, \um@usv@itvarrho}{\um@usv@bfvarrho}
1084   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi, \um@usv@itvarpi}{\um@usv@bfvarpi}
1085 }

```

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

***ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz***

`$\mathbffrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbffrak{abcdefghijklmnopqrstuvwxyz}$ \\`

```

1086 \cs_new:Npn \um_config_mathbffrak: {
1087   \um_set_mathalphabet_numbers:Nnn{\mathbffrak}{\um@usv@num}{\um@usv@bfnun}
1088   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin, \um@usv@itLatin, \um@usv@frakLa}
1089   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin, \um@usv@itlatin, \um@usv@frakla}
1090 }

```

8.1.5 Bold script or calligraphic: `\mathbfscr`

***ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz***

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

```

1091 \cs_new:Npn \um_config_mathbfscr: {
1092   \um_set_mathalphabet_numbers:Nnn{\mathbfscr}{\um@usv@num}{\um@usv@bfnun}
1093   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin, \um@usv@itLatin}{\um@usv@bfscrL}
1094   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin, \um@usv@itlatin}{\um@usv@bfscr1}
1095 }

```

8.1.6 Bold sans serif: `\mathbf{f}`

0123456789
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
 ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ
 αβγδεζηθικλμνξοπρστυφχψω εδκφορτ

```
\setmathfont{STIXGeneral-Bold}
$\mathbf{f}{0123456789}$ \\\
$\mathbf{f}{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\\
$\mathbf{f}{abcdefghijklmnopqrstuvwxyz}$ \\\
$\mathbf{f}{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad
$\mathbf{f}{Θ}$ \\\
$\mathbf{f}{αβγδεζηθικλμνξοπρστυφχψω}$\quad
$\mathbf{f}{εδκφορτ}$ \\\
```

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

```
1096 \cs_new:Npn \um_config_mathbf f: {
1097   \um_set_mathalphabet_numbers:Nnn{\mathbf f}{\um@sv@num}{\um@sv@b fnum}
1098   \um_set_mathalphabet_latin:Nnn{\mathbf f}{\um@sv@upLatin,\um@sv@i tLatin}{\um@sv@b f flat
1099   \um_set_mathalphabet_latin:Nnn{\mathbf f}{\um@sv@uplatin,\um@sv@i tlatin}{\um@sv@b f flat
1100   \um_set_mathalphabet_greek:Nnn{\mathbf f}{\um@sv@upGreek,\um@sv@i tGreek}{\um@sv@b f gree
1101   \um_set_mathalphabet_greek:Nnn{\mathbf f}{\um@sv@upgreek,\um@sv@i tgreek}{\um@sv@b f gree
```

Others:

```
1102   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@varTheta,\um@sv@i tvarTheta}"1D767}
1103   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@Nabla,\um@sv@i tNabla}"1D76F}
1104   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@partial,\um@sv@i tpartial}"1D789}
1105   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@varepsilon,\um@sv@i tvarepsilon}"1D78A}
1106   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@vartheta,\um@sv@i tvartheta}"1D78B}
1107   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@varkappa,\um@sv@i tvarkappa}"1D78C}
1108   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@varphi,\um@sv@i tvarphi}"1D78D}
1109   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@varrho,\um@sv@i tvarrho}"1D78E}
1110   \um_set_mathalphabet_char:Nnn{\mathbf f}{\um@sv@varpi,\um@sv@i tvarpi}"1D78F}
1111 }
```

8.1.7 Bold upright sans serif: `\mathbf{fup}`

0123456789
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
 ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ
 αβγδεζηθικλμνξοπρστυφχψω εδκφορτ

```
\setmathfont{STIXGeneral-Bold}
$\mathbf{fup}{0123456789}$ \\\
$\mathbf{fup}{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\\
$\mathbf{fup}{abcdefghijklmnopqrstuvwxyz}$ \\\
$\mathbf{fup}{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad
$\mathbf{fup}{Θ}$ \\\
$\mathbf{fup}{αβγδεζηθικλμνξοπρστυφχψω}$\quad
$\mathbf{fup}{εδκφορτ}$ \\\
```

Numbers (always upright) and letters:

```
1112 \cs_new:Npn \um_config_mathbf fup: {
1113   \um_set_mathalphabet_numbers:Nnn{\mathbf fup}{\um@sv@num}{\um@sv@b fnum}
1114   \um_set_mathalphabet_latin:Nnn{\mathbf fup}{\um@sv@upLatin,\um@sv@i tLatin}{\um@sv@b f flat
1115   \um_set_mathalphabet_latin:Nnn{\mathbf fup}{\um@sv@uplatin,\um@sv@i tlatin}{\um@sv@b f flat
```

```

1116 \um_set_mathalphabet_greek: Nnn{\mathbfs fup}{\um@usv@upGreek, \um@usv@itGreek}{\um@usv@bfs fg
1117 \um_set_mathalphabet_greek: Nnn{\mathbfs fup}{\um@usv@upgreek, \um@usv@itgreek}{\um@usv@bfs fg

```

Others:

```

1118 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@varTheta, \um@usv@itvarTheta}{1D767}
1119 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@Nabla, \um@usv@itNabla}{1D76F}
1120 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@partial, \um@usv@itpartial}{1D789}
1121 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@varepsilon, \um@usv@itvarepsilon}{1D78A}
1122 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@vartheta, \um@usv@itvartheta}{1D78B}
1123 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@varkappa, \um@usv@itvarkappa}{1D78C}
1124 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@varphi, \um@usv@itvarphi}{1D78D}
1125 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@varrho, \um@usv@itvarrho}{1D78E}
1126 \um_set_mathalphabet_char: Nnn{\mathbfs fup}{\um@usv@varpi, \um@usv@itvarpi}{1D78F}
1127 }

```

8.1.8 Bold italic sans serif: `\mathbfs fit`

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ
αβγδεζηθικλμνξοπρστυφχψω εδκφρϖ

```

\setmathfont{STIXGeneral-BoldItalic}
$\mathbfs fit{0123456789}$ \\\
$\mathbfs fit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\\
$\mathbfs fit{abcdefghijklmnopqrstuvwxyz}$ \\\
$\mathbfs fit{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad
$\mathbfs fit{␣}$ \\\
$\mathbfs fit{αβγδεζηθικλμνξοπρστυφχψω}$\quad
$\mathbfs fit{␣␣␣␣␣}$ \\\

```

```

1128 \cs_new:Npn \um_config_mathbfs fit: {
1129 \um_set_mathalphabet_numbers: Nnn{\mathbfs fit}{\um@usv@num}{\um@usv@bfnun}
1130 \um_set_mathalphabet_latin: Nnn{\mathbfs fit}{\um@usv@upLatin, \um@usv@itLatin}{\um@usv@bfs fi
1131 \um_set_mathalphabet_latin: Nnn{\mathbfs fit}{\um@usv@uplatin, \um@usv@itlatin}{\um@usv@bfs fi
1132 \um_set_mathalphabet_greek: Nnn{\mathbfs fit}{\um@usv@upGreek, \um@usv@itGreek}{\um@usv@bfs fi
1133 \um_set_mathalphabet_greek: Nnn{\mathbfs fit}{\um@usv@upgreek, \um@usv@itgreek}{\um@usv@bfs fi

```

Other symbols:

```

1134 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@varTheta}{1D7A1}
1135 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@Nabla, \um@usv@itNabla}{\um@usv@bfs fitNa
1136 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@partial, \um@usv@itpartial}{\um@usv@bfs fi
1137 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@varepsilon, \um@usv@itvarepsilon}{1D7C4}
1138 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@vartheta, \um@usv@itvartheta}{1D7C5}
1139 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@varkappa, \um@usv@itvarkappa}{1D7C6}
1140 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@varphi, \um@usv@itvarphi}{1D7C7}
1141 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@varrho, \um@usv@itvarrho}{1D7C8}
1142 \um_set_mathalphabet_char: Nnn{\mathbfs fit}{\um@usv@varpi, \um@usv@itvarpi}{1D7C9}
1143 }

```

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 \LaTeX ‘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.

```
1144 \begingroup
1145   \char_make_other:N \^
1146   \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1147     \lowercase{
1148       \scantokens{\global\let#1=^^^^^#2}
1149     }
1150   }
```

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

```
1151   \gdef\um@scanactivedef"#1\@nil#2{
1152     \lowercase{
1153       \tl_rescan:nn{
1154         \char_make_math_superscript:N\^
1155       }{
1156         \global\def^^^^^#1{#2}
1157       }
1158     }
1159   }
1160 \endgroup
```

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

```
1161 \begingroup
1162   \def\UnicodeMathSymbol#1#2#3#4{
1163     \um@scancharlet#2=#1\@nil
1164   }
1165   \@input{unicode-math-table.tex}
1166 \endgroup
```

9 Epilogue

Lots of little things to tidy up.

9.0.1 Primes

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

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

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: xʹ
```

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: `xʹ` and `xʹ`. 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 & `\primetripel`.
- Ditto pcount=4 & `\primequadruple`.
- If pcount>4 or the glyph doesn’t exist, insert pcount `\primes` with `\primekern` between each.

```
1167 \muskip_new: N \g_um_primekern_muskip
1168 \muskip_gset: Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1169 \num_new: N \l_um_primecount_num
```



```

1170 \cs_new:Nn \um_nprimes:n {
1171   \primesingle
1172   \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1173 }
1174 \cs_new:Nn \um_nprimes_select:n {
1175   \prg_case_int:nnn {#1}{
1176     {1} { \primesingle }
1177     {2} {
1178       \um_glyph_if_exist:NTF {"2033} {\primedouble} {\um_nprimes:n {#1}}
1179     }
1180     {3} {
1181       \um_glyph_if_exist:NTF {"2034} {\primetriple} {\um_nprimes:n {#1}}
1182     }
1183     {4} {
1184       \um_glyph_if_exist:NTF {"2057} {\primequadruple} {\um_nprimes:n {#1}}
1185     }
1186   }{
1187     \um_nprimes:n {#1}
1188   }
1189 }

```

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?

```

1190 \cs_new:Nn \um_scanprime: {
1191   \bgroup
1192   \num_zero:N \l_um_primecount_num
1193   \um_scanprime_collect:
1194 }
1195 \cs_new:Nn \um_scanprime_collect: {
1196   \num_incr:N \l_um_primecount_num
1197   \peek_charcode_remove:NTF ' ' {
1198     \um_scanprime_collect:
1199   }{
1200     \peek_meaning_remove:NTF \um_scanprime: {
1201       \um_scanprime_collect:
1202     }{
1203       \peek_charcode_remove:NTF ^^^^2032 {
1204         \um_scanprime_collect:
1205       }{
1206         \um_nprimes_select:n {\l_um_primecount_num}
1207         \egroup
1208       }
1209     }
1210 }

```

```

1211 }
1212 \cs_set_eq:NN \prime \um_scanprime:
1213 \group_begin:
1214   \char_make_active:N \'
1215   \char_make_active:n {"2032}
1216   \cs_gset_eq:NN ' \um_scanprime:
1217   \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1218 \group_end:

```

9.0.2 Unicode radicals

Undo the damage made to `\sqrt`:

```

1219 \DeclareRobustCommand\sqrt{\@i fnextchar[\@sqrt\sqrtsign}

```

`\r@@t #1` : A mathstyle (for `\mathpalette`)
`#2` : Leading superscript for the sqrt sign
 A re-implementation of \LaTeX 's hard-coded n-root sign using the appropriate `\fontdimens`.

```

1220 \def\r@@t#1#2{
1221   \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
1222   \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1223   \raise \dimexpr(
1224     \um@fontdimen@percent{65}{\um@font}\ht\z@-
1225     \um@fontdimen@percent{65}{\um@font}\dp\z@
1226   )\relax
1227   \copy \rootbox
1228   \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1229   \box \z@
1230 }

```

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:

```

1231 \prop_new:N \g_um_supers_prop
1232 \prop_new:N \g_um_subs_prop
1233 \cs_generate_variant:Nn \prop_gput:Nnn {Nxn}

```

```

1234 \cs_generate_variant:Nn \prop_get:NnN {cxN}
1235 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}
1236
1237 \group_begin:
1238
1239 % Populate a property list with superscript characters; their mean-
1240 % ing as their key,
1241 % for reasons that will become apparent soon, and their replace-
1242 % ment as each key's value.
1243 % Then make the superscript active and bind it to the scanning function.
1244 %
1245 % \cs{scantokens} makes this process much simpler since we can acti-
1246 % vate the char
1247 % and assign its meaning in one step.
1248 \cs_set:Nn \um_setup_active_superscript:nn {
1249   \prop_gput:Nxn \g_um_supers_prop {\meaning #1} {#2}
1250   \char_make_active:n {\`#1}
1251   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1252   \scantokens{
1253     \cs_gset:Npn #1 {
1254       \tl_set:Nn \l_um_ss_chain_tl {#2}
1255       \cs_set_eq:NN \um_sub_or_super:n \sp
1256       \tl_set:Nn \l_um_tmpa_tl {supers}
1257       \um_scan_ssript:
1258     }
1259   }
1260 }
1261
1262 \um_setup_active_superscript:nn {^^^2071} {i}
1263 \um_setup_active_superscript:nn {^^^207f} {n}
1264 \um_setup_active_superscript:nn {^^^2070} {0}
1265 \um_setup_active_superscript:nn {^^^00b9} {1}
1266 \um_setup_active_superscript:nn {^^^00b2} {2}
1267 \um_setup_active_superscript:nn {^^^00b3} {3}
1268 \um_setup_active_superscript:nn {^^^2074} {4}
1269 \um_setup_active_superscript:nn {^^^2075} {5}
1270 \um_setup_active_superscript:nn {^^^2076} {6}
1271 \um_setup_active_superscript:nn {^^^2077} {7}
1272 \um_setup_active_superscript:nn {^^^2078} {8}
1273 \um_setup_active_superscript:nn {^^^2079} {9}
1274 \um_setup_active_superscript:nn {^^^207a} {+}
1275 \um_setup_active_superscript:nn {^^^207b} {-}
1276 \um_setup_active_superscript:nn {^^^207c} {=}
1277 \um_setup_active_superscript:nn {^^^207d} {(}
1278 \um_setup_active_superscript:nn {^^^207e} {)}

```

```

1277 % Ditto above.
1278 \cs_set:Nn \um_setup_active_subscript:nn {
1279   \prop_gput:Nxn \g_um_subs_prop {\meaning #1} {#2}
1280   \char_make_active:n {\`#1}
1281   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1282   \scantokens{
1283     \cs_gset:Npn #1 {
1284       \tl_set:Nn \l_um_ss_chain_tl {#2}
1285       \cs_set_eq:NN \um_sub_or_super:n \sb
1286       \tl_set:Nn \l_um_tmpa_tl {subs}
1287       \um_scan_sscript:
1288     }
1289   }
1290 }
1291
1292 \um_setup_active_subscript:nn {^^^2080} {0}
1293 \um_setup_active_subscript:nn {^^^2081} {1}
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 vari-
1334   ous \cs{peek_} functions.
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 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`:

```

1378 \def\operator@font{\um_setup_mathup:}
1379 \def\to{\rightarrow}
1380 \def\le{\leq}
1381 \def\ge{\geq}

```

`\mathcal`

```

1382 \def\mathcal{\mathscr}

```

`\mathrm`

```

1383 \def\mathrm{\mathup}

```

Overriding amsmath definitions:

```

1384 \AtBeginDocument{
1385   \def\@cdots{\mathinner{\cdots}}
1386 }

```

Interaction with beamer:

```

1387 \AtBeginDocument{
1388   \@ifpackageloaded{beamer}{
1389     \ifbeamer@suppressreplacements\else
1390       \PackageWarningNoLine{unicode-math}{
1391         Disabling~ beamer's~ math~ setup.^^J
1392         Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1393       }
1394     \beamer@suppressreplacementstrue
1395   \fi
1396 }{}
1397 }

```

The end.

```

1398 \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 X_YTeX, 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 T_EX macro names.

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

A ‘fence’ defined by the STIX table is something like `\vert`; in X_ET_EX 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 `{T1}{1mr}{m}{n}`.

Maths symbol fonts Fonts for symbols: α , \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 $ABC-xyz$, $\mathfrak{ABC}-\mathcal{XZ}$, 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 TeX's `\delimiter`/`\radical` primitives, which are re-designed in XeTeX. The syntax used in LaTeX'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 XeTeX.

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

XeTeX math font dimensions

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

\fontdimen	Dimension name	Description
10	SCRIPTPERCENTSCALEDOWN	Percentage of scaling down for script level 1. Suggested value: 80%.
11	SCRIPTSCRIPTPERCENTSCALE-DOWN	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	DELIMITEDSUBFORMULAMIN-HEIGHT	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height \times 1.5.
13	DISPLAYOPERATORMINHEIGHT	Minimum height of n-ary operators (such as integral and summation) for formulas in display mode.
14	MATHLEADING	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 (os2.sTypoAscender + os2.sTypoLineGap – MathLeading) or with ink going below os2.sTypoDescender will result in increasing line height.
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: $/5$ x-height.

\fontdimen	Dimension name	Description
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: $4 \times$ default rule thickness.
26	SUPERSCRIPBOTTOMMAX-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	SPACEAFTERSCRIP	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.

\fontdimen	Dimension name	Description
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	STACKTOPDISPLAYSTYLESHIFTUP	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	STACKBOTTOMDISPLAYSTYLESHIFTDOWN	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×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×default rule thickness.
38	STRETCHSTACKTOPSHIFTUP	Standard shift up applied to the top element of the stretch stack.
39	STRETCHSTACKBOTTOMSHIFTDOWN	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	FRACTIONNUMERATORDISPLAYSTYLESHIFTUP	Standard shift up applied to the numerator in display style. Suggested: StackTopDisplayStyleShiftUp.

\fontdimen	Dimension name	Description
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
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.

\fontdimen	Dimension name	Description
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.
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{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}
10 \def\upgreek{αβγδεζηθικλμνξοπρστυφχψω}
11 \def\itLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
12 \def\itlatin{abcdefghijklmnopqrstuvwxyz}
13 \def\itGreek{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}
14 \def\itgreek{αβγδεζηθικλμνξοπρστυφχψω}
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=225FF99]{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 \end{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{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΨΣΤΥΦΧΨΩ}
45 \def\upgreek{αβγδεζηθικλμνξοπρςστυφχψω}
46 \def\itLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
47 \def\itlatin{abcdefghijklmnopqrstuvwxyz}
48 \def\itGreek{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΨΣΤΥΦΧΨΩ}
49 \def\itgreek{αβγδεζηθικλμνξοπρςστυφχψω}
50 \def\bfulatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
51 \def\bfulatin{abcdefghijklmnopqrstuvwxyz}
52 \def\bfulatin{αβγδεζηθικλμνξοπρςστυφχψω}
53 \def\bfulatin{αβγδεζηθικλμνξοπρςστυφχψω}
54 \def\bfitLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
55 \def\bfitlatin{abcdefghijklmnopqrstuvwxyz}
56 \def\bfitgreek{αβγδεζηθικλμνξοπρςστυφχψω}
57 \def\bfitgreek{αβγδεζηθικλμνξοπρςστυφχψω}
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
\'	1214
\-	9
\:::	677
\::f	677
\::n	677
\=	10
\@DeclareMathDelimiter	276
\@DeclareMathSizes	241
\@backslashchar	841
\@begindocumenthook	8
\@ccclvi	301
\@cdots	1385
\@elt	548–552, 555, 559, 561
\@empty	352, 353, 390, 476, 593, 600, 615, 636, 641
\@ifnextchar	1219
\@ifpackageloaded	7, 1388
\@iii	599, 600, 602, 604, 607, 612
\@input	407, 1165
\@marker	612, 631
\@nil	310, 454, 612, 625–628, 1146, 1151, 1163
\@preamblecmds	285
\@sqrt	1219
\@tempa	125, 168, 191, 209, 227, 363, 379, 591, 603, 627, 629, 641
\@tempb	125, 126, 168, 169, 191, 192, 209, 210, 227, 228, 591, 592, 630, 631, 636
\@tempswafalse	601
\@tempswatruetrue	605, 608, 633, 638, 643, 648
\@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@upNablatrue	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, 1145, 1154
Numbers	
\0	29
_	17–20, 23–32
A	
\A	30
\a	31
\addnolimits	<u>554</u>
\addtoversion	248
\advance	30, 77
\alloc@	301
\Alpha	704
\alpha	729
\alpha@elt	252
\alpha@list	251
\AtBeginDocument	702, 1384, 1387
\awint	550

B	
<code>\B</code>	932
<code>\beamer@suppressreplacementstrue</code>	1394
<code>\begin</code>	19, 66
<code>\begingroup</code> ...	307, 558, 859, 1144, 1161
<code>\Beta</code>	705
<code>\beta</code>	730, 1316
<code>\bfitGreek</code>	56
<code>\bfitgreek</code>	57
<code>\bfitLatin</code>	54
<code>\bfitlatin</code>	55
<code>\bfupGreek</code>	52
<code>\bfupgreek</code>	53
<code>\bfupLatin</code>	50
<code>\bfuplatin</code>	51
<code>\bgroup</code>	1191
<code>\bool_if: NTF</code>	734, 753, 759, 764
<code>\bool_if: nTF</code>	1346
<code>\bool_new: N</code>	28
<code>\bool_set_false: N</code> ...	137, 161, 165, 229
<code>\bool_set_true: N</code>	149, 231
<code>\box</code>	32, 33, 79, 80, 1229
C	
<code>\C</code>	921, 947
<code>\c_group_begin_token</code>	1347
<code>\c_peek_true_remove_next_tl</code> ...	1337
<code>\c_space_token</code>	1348
<code>\cdots</code>	1385
<code>\cdp@elt</code>	239
<code>\cdp@list</code>	238
<code>\char_make_active: N</code>	1214
<code>\char_make_active: n</code>	308, 1215, 1247, 1280
<code>\char_make_math_superscript: N</code> ..	1154
<code>\char_make_other: N</code>	1145
<code>\chardef</code>	301
<code>\Chi</code>	726
<code>\chi</code>	755, 1320
<code>\cirfnint</code>	550
<code>\clist_map_inline: Nn</code>	533
<code>\clist_map_inline: nn</code> ...	459, 528, 535
<code>\clist_map_variable: NNn</code>	599
<code>\clist_map_variable: nNn</code> .	656, 679, 685
<code>\copy</code>	1227
<code>\cs</code>	1243, 1333
<code>\cs_generate_variant: Nn</code> ...	1233–1235
<code>\cs_gset: cpn</code>	315, 325
<code>\cs_gset: Npn</code> . .	328, 333, 1146, 1250, 1283
<code>\cs_gset: Npx</code>	338
<code>\cs_gset_eq: NN</code>	1216, 1217
<code>\cs_if_exist: cF</code>	857
<code>\cs_if_exist: cT</code>	842
<code>\cs_new: Nn</code>	414, 426, 453, 458, 463, 469, 678, 684, 693, 696, 699, 834, 856, 869, 1170, 1174, 1190, 1195, 1325, 1335, 1364
<code>\cs_new: Npn</code> .	887, 902, 918, 930, 945, 954, 959, 964, 969, 1046, 1066, 1086, 1091, 1096, 1112, 1128, 1345
<code>\cs_set: cpn</code>	858
<code>\cs_set: Nn</code>	198, 216, 347, 418, 421, 475, 527, 532, 655, 665, 668, 671, 674, 769, 780, 790, 797, 806, 815, 847, 850, 1245, 1278
<code>\cs_set: Npn</code>	677
<code>\cs_set_eq: cN</code>	851
<code>\cs_set_eq: NN</code>	393–396, 400–403, 1212, 1252, 1285
<code>\cs_set_protected: cpn</code>	861
<code>\cs_to_str: N</code>	325, 543
<code>\csname</code>	17, 18, 61–64, 304, 310, 313, 316, 348, 355, 471, 543, 835, 863
D	
<code>\DeclareDocumentCommand</code>	350
<code>\DeclareMathAccent</code>	269
<code>\DeclareMathAlphabet</code>	265
<code>\DeclareMathDelimiter</code>	274
<code>\DeclareMathRadical</code>	280
<code>\DeclareMathSizes</code>	240
<code>\DeclareMathSymbol</code>	271
<code>\DeclareMathVersion</code>	245, 357
<code>\DeclareRobustCommand</code>	1219
<code>\DeclareSymbolFont</code>	258, 405
<code>\DeclareSymbolFontAlphabet</code>	282
<code>\DeclareSymbolFontAlphabet@</code>	283
<code>\def</code>	7–15, 29–69, 71–123, 287, 290, 301, 303, 305, 356, 547, 555, 557, 559, 566, 568, 625, 627–630, 704–733, 736–752, 755–758, 761–763, 766, 767, 1156, 1162, 1220, 1378–1383, 1385
<code>\define@choicekey</code>	124, 168, 191, 209, 227, 591

\define@cmdkey	587–590	\f@size	355
\define@key	570	\fi . 166, 189, 196, 207, 214, 225, 232,	
\define@mathalphabet	246	298, 299, 320, 341–345, 389, 404,	
\define@mathgroup	247	436, 451, 485, 493, 498, 505, 522,	
\Delta	707	523, 525, 538, 562, 578, 594, 609,	
\delta	732	610, 613, 619, 621, 622, 634, 639,	
\dimen	29–31, 76–78	644, 649–653, 795, 804, 813, 832,	
\dimexpr	288, 381, 1223	1008, 1015, 1022, 1041, 1044, 1395	
\do	285	\fi:	854
\documentclass	2, 37	\fint	550
\dorestore@version	255	\font	380
\dp	30, 77, 1225	\fontdimen	288, 381, 1222, 1228
E		\fontname	841
\E	933	G	
\e	941	\g	942
\edef	31, 78, 363, 626, 627	\g@addto@macro	542, 616, 618
\egroup	1207	\g_um_primekern_muskip	1167, 1168, 1172
\else	203, 221, 293, 296,	\g_um_subs_prop	1232, 1279
322, 326, 331, 336, 339, 383, 397,		\g_um_supers_prop	1231, 1246
433, 446, 480, 488, 491, 496, 502,		\g_um_texgreek_bool . 28, 137, 149,	
514, 524, 560, 573, 606, 611, 617,		161, 165, 229, 231, 734, 753, 759, 764	
635, 640, 645, 793, 801, 810, 824,		\Gamma	706
1003, 1006, 1012, 1019, 1032, 1389		\gamma	731, 1317
\else:	854	\gdef	1151
\encodingdefault	406, 848	\ge	1381
\end	34, 81	\geq	1381
\endcsname . . 17, 18, 61–64, 304, 310,		\get@cdp	264
313, 316, 348, 355, 471, 543, 835, 863		\glb@currsiz	351
\endgroup	311, 564, 859, 1160, 1166	\global	309, 312, 329, 330, 334,
\Epsilon	708	335, 340, 593, 1148, 1156, 1248, 1281	
\epsilon	733	\group@elt	260
\Eta	710	\group@list	259
\eta	737	\group_align_safe_begin:	1339
\etex_iffontchar:D	854	\group_align_safe_end:	1338
\ExecuteOptionsX	234	\group_begin:	1213, 1237
\exp_args:Nnff	677, 680, 687	\group_end:	1218, 1322
\exp_args:No	529, 536	H	
\exp_not:n	1336, 1338	\H	922, 935, 948
\expandafter		\h	800, 803, 905
304, 312, 317, 319, 323, 542, 555,		\hbox	1221
602, 604, 607, 612, 626, 627, 631, 863		\hoffset	23, 70
\ExplSyntaxOff	1398	\ht	29, 76, 1224
\ExplSyntaxOn	6	I	
F		\I	936, 949
\F	934		

\backslash if@tempswa	614	L	
\backslash if@um@bfliteral	21, 437, 486, 973	\backslash L	937
\backslash if@um@bfupGreek	22, 499, 1016	\backslash l_peek_false_tl	1338, 1350, 1376
\backslash if@um@bfupgreek	23, 506, 1023	\backslash l_peek_token	1347, 1348, 1367, 1371
\backslash if@um@bfupLatin	24, 489, 1004	\backslash l_peek_true_aux_tl	1336
\backslash if@um@bfuplatin	25, 494, 1009	\backslash l_peek_true_tl	1337, 1374
\backslash if@um@fontspec@feature	14, 571	\backslash l_um_inc_num	686, 688, 689
\backslash if@um@literal	16, 428, 478	\backslash l_um_incr_num	657, 659, 661
\backslash if@um@ot@math@	15	\backslash l_um_input_num	
\backslash if@um@upGreek	17, 807		656, 659, 679, 681, 685, 688
\backslash if@um@upgreek	18, 816	\backslash l_um_primecount_num	
\backslash if@um@upLatin	19, 791		1169, 1192, 1196, 1206
\backslash if@um@uplatin	20, 798	\backslash l_um_script_features_tl	358, 369
\backslash if@um@upNabla	26, 199	\backslash l_um_script_font_tl	360, 368
\backslash if@um@uppartial	27, 217	\backslash l_um_ss_chain_tl	1251, 1284, 1329, 1373
\backslash ifbeamer@suppressreplacements	1389	\backslash l_um_sscript_features_tl	359, 373
\backslash ifcase	126, 169, 192, 210, 228, 592	\backslash l_um_sscript_font_tl	361, 372
\backslash ifdim	381	\backslash l_um_tmpa_tl	1253, 1286, 1366, 1370
\backslash ifin@	318, 324	\backslash l_um_tmpb_tl	1372, 1373
\backslash ifnum	534, 632, 637, 642, 646, 647	\backslash Lambda	714
\backslash ifx	291, 294,	\backslash lambda	741
	306, 327, 332, 337, 390, 476, 560,	\backslash le	1380
	600, 603, 604, 607, 615, 631, 636, 641	\backslash left	567
\backslash iiint	548	\backslash left@primitive	567, 568
\backslash iiint	548	\backslash leq	1380
\backslash iint	548	\backslash let	56, 70, 302, 351–353, 567, 593, 1148
\backslash in@	317, 323	\backslash linewidth	16, 60
\backslash init@restore@version	254	\backslash lowercase	1147, 1152
\backslash int	548	\backslash lowint	552
\backslash intBar	550	M	
\backslash intbar	550	\backslash M	938
\backslash intcap	552	\backslash m@th	1221
\backslash intclockwise	549	\backslash makebox	16–18, 60–64
\backslash intcup	552	\backslash mathaccent	337
\backslash intlarhk	551	\backslash mathalpha	465–467, 544, 660
\backslash intx	551	\backslash mathalphabet	58, 61, 62
\backslash Iota	712	\backslash mathbb	919–928
\backslash iota	739	\backslash mathbf	58, 970–972, 974–1002,
\backslash itGreek	13, 48		1005, 1007, 1010, 1011, 1013,
\backslash itgreek	14, 49		1014, 1017, 1018, 1020, 1021,
\backslash itLatin	11, 46		1024–1031, 1033–1040, 1042, 1043
\backslash itlatin	12, 47	\backslash mathbffrak	1087–1089
K		\backslash mathbfit	1047–1064
\backslash Kappa	713	\backslash mathbfscr	1092–1094
\backslash kappa	740	\backslash mathbfsf	1097–1110
\backslash kern	1222, 1228	\backslash mathbfsfit	1129–1142

<code>\new@symbolfont</code>	261	<code>\Pi</code>	719
<code>\newcommand</code>	541, 554, 569, 598, 703	<code>\pi</code>	746
<code>\newcounter</code>	13	<code>\pointint</code>	551
<code>\newfam</code>	302	<code>\prg_case_int:nnn</code>	1175
<code>\newif</code>	14–27	<code>\prg_do_nothing:</code>	851
<code>\newmathalphabet</code>	242	<code>\prg_new_conditional:Nnn</code>	853
<code>\newmathalphabet@@</code>	243	<code>\prg_replicate:nn</code>	1172
<code>\newmathalphabet@@@</code>	244	<code>\prg_return_false:</code>	854
<code>\noexpand</code>	363, 561	<code>\prg_return_true:</code>	854
<code>\nolimits</code>	319	<code>\prg_stepwise_variable:nnnNn</code> 657, 686	
<code>\non@alpherr</code>	863	<code>\prime</code>	1212
<code>\npolint</code>	551	<code>\primedouble</code>	1178
<code>\Nu</code>	716	<code>\primequadruple</code>	1184
<code>\nu</code>	743	<code>\primesingle</code>	464, 1171, 1172, 1176
<code>\num_incr:N</code>	1196	<code>\primetripel</code>	1181
<code>\num_new:N</code>	1169	<code>\process@table</code>	256
<code>\num_zero:N</code>	1192	<code>\ProcessOptionsX</code>	235
<code>\number</code>	681, 688, 689	<code>\prop_get:cxN</code>	1369
<code>\numexpr</code>	637,	<code>\prop_get:NnN</code>	1234
642, 646, 647, 659, 661, 681, 688, 689		<code>\prop_gput:Nnn</code>	1233
		<code>\prop_gput:Nxn</code>	1246, 1279
O		<code>\prop_if_in:cxTF</code>	1365
<code>\o</code>	943	<code>\prop_if_in:NnTF</code>	1235
<code>\oiint</code>	548	<code>\prop_new:N</code>	1231, 1232
<code>\oint</code>	548	<code>\protect</code>	577
<code>\ointctrcllockwise</code>	549	<code>\providecommand</code>	58
<code>\Omega</code>	728	<code>\ProvidesPackage</code>	1
<code>\omega</code>	757	<code>\Psi</code>	727
<code>\Omicron</code>	718	<code>\psi</code>	756
<code>\omicron</code>	745		
<code>\operator@font</code>	1378	Q	
<code>\or</code>	138,	<code>\Q</code>	925
150, 162, 175, 181, 187, 194, 212, 230			
P		R	
<code>\P</code>	924	<code>\R</code>	926, 939, 950
<code>\PackageError</code>	574	<code>\r@t</code>	1220
<code>\PackageInfo</code>	392	<code>\raise</code>	1223
<code>\PackageWarningNoLine</code> ..	384, 840, 1390	<code>\relax</code>	9, 10, 126,
<code>\papersize</code>	31, 32, 78, 79	169, 192, 210, 228, 288, 306, 309,	
<code>\parindent</code>	21, 68	314, 325, 327–330, 332–335, 337,	
<code>\peek_after:NN</code>	1340	338, 340, 348, 351, 380, 381, 534,	
<code>\peek_charcode_remove:NTF</code> .	1197, 1203	592, 604, 607, 631, 632, 637, 642,	
<code>\peek_meaning_remove:NTF</code>	1200	646, 647, 659, 661, 681, 688, 689, 1226	
<code>\Phi</code>	725	<code>\removenolimits</code>	557
<code>\phi</code>	752, 1319	<code>\RequirePackage</code>	3–5
		<code>\restore@mathversion</code>	253
		<code>\Rho</code>	720

\backslash rho	747, 1318	\backslash tau	750
\backslash rightarrow	1379	\backslash testmath	15, 25–28, 59, 72–75
\backslash rootbox	1227	\backslash tf@size	366, 367
\backslash rppolint	550	\backslash the	31, 78
S		\backslash Theta	711
\backslash sb	1285	\backslash theta	738
\backslash scan_stop: ...	472, 473, 854, 1248, 1281	\backslash theum@fam	399
\backslash scantokens	1148, 1249, 1282	\backslash thinmuskip	1168
\backslash scpolint	551	\backslash tl_map_inline:nn	236
\backslash scriptscriptstyle	294	\backslash tl_put_right:NV	1373
\backslash scriptstyle	291	\backslash tl_remove_in:Nn	8, 285
\backslash set@@mathdelimiter	279	\backslash tl_rescan:nn	1153
\backslash set@mathaccent	270	\backslash tl_set:Nn	200–202, 204–206, 218–220, 222–224, 358–361, 391, 1251, 1253, 1284, 1286
\backslash set@mathchar	272	\backslash tl_set:Nx	399, 1336, 1338
\backslash set@mathdelimiter	278	\backslash tl_set_eq:NN	1337
\backslash set@mathsymbol	273	\backslash to	1379
\backslash setbox	24, 32, 71, 79, 1221	\backslash token_if_eq_catcode_p:NN	1347
\backslash setkeys	362	\backslash token_if_eq_meaning_p:NN	1348
\backslash setmainfont	5, 40	\backslash ttdefault	848
\backslash SetMathAlphabet	267, 848	U	
\backslash SetMathAlphabet@	268	\backslash um@addto@mathmap	529, 536, <u>541</u>
\backslash setmathfont	20, 67, <u>350</u> , 577	\backslash um@backslash	603, 626
\backslash SetSymbolFont	262	\backslash um@char@num@range	353, 533, 615, 616, 618
\backslash SetSymbolFont@	263	\backslash um@char@range	352, 390, 476, 593, 596, 599
\backslash sf@size	367, 371	\backslash um@firstchar	602, 627
\backslash shipout	33, 80	\backslash um@firstof	625–627
\backslash Sigma	722	\backslash um@font	292, 295, 380, 381, 841, 854, 1222, 1224, 1225, 1228
\backslash sigma	749	\backslash um@fontdimen@percent	<u>287</u> , 292, 295, 1224, 1225
\backslash sp	1252	\backslash um@mathsymbol	<u>303</u> , 419
\backslash space	840, 863	\backslash um@mversion	356, 357
\backslash special	32, 79	\backslash um@nolimits	317, <u>547</u> , 555, 563
\backslash sqint	551	\backslash um@parse@range	612, <u>628</u>
\backslash sqrt	566, 1219	\backslash um@parse@term	422, 454, <u>598</u>
\backslash sqrtsign	1219, 1221	\backslash um@radicals	323, <u>566</u>
\backslash std@equal	10	\backslash um@resolve@greek	<u>702</u>
\backslash std@minus	9	\backslash um@scaled@apply	<u>290</u> , 1222, 1228
\backslash stepcounter	398	\backslash um@scanactivatedef	310, <u>1144</u>
\backslash string	310, 313, 315, 316, 626, 627	\backslash um@scancharlet	<u>1144</u> , 1163
\backslash strip@pt	288	\backslash um@set@mathsymbol	304, <u>305</u>
\backslash subscriptone	466	\backslash um@usv@bbLatin	39, 920
\backslash subscripttwo	467	\backslash um@usv@bbLatin	40, 928
\backslash subscriptzero	465	\backslash um@usv@bbnum	38, 919
\backslash sumint	549		
T			
\backslash Tau	723		

$\backslash\mathrm{um@usv@b fDi gamma}$	87, 985	$\backslash\mathrm{um@usv@b fscr latin}$	66, 1094
$\backslash\mathrm{um@usv@b fdigamma}$	94, 993	$\backslash\mathrm{um@usv@b fs fGreek}$	71, 1100, 1116
$\backslash\mathrm{um@usv@b ffrak Latin}$	63, 1088	$\backslash\mathrm{um@usv@b fs fgreek}$	72, 1101, 1117
$\backslash\mathrm{um@usv@b ffrak latin}$	64, 1089	$\backslash\mathrm{um@usv@b fs fitGreek}$	75, 1132
$\backslash\mathrm{um@usv@b fGreek}$	57, 500, 503, 785, 978, 1017, 1054, 1070, 1074	$\backslash\mathrm{um@usv@b fs fitgreek}$	76, 1133
$\backslash\mathrm{um@usv@b fgreek}$	58, 507, 515, 786, 980, 1024, 1055, 1071, 1075	$\backslash\mathrm{um@usv@b fs fitLatin}$	73, 1130
$\backslash\mathrm{um@usv@b fitGreek}$	61, 500, 503, 787, 979, 1020, 1050, 1054	$\backslash\mathrm{um@usv@b fs fitlatin}$	74, 1131
$\backslash\mathrm{um@usv@b fitgreek}$	62, 507, 515, 788, 981, 1033, 1051, 1055	$\backslash\mathrm{um@usv@b fs fitNabla}$	117, 206, 441, 448, 1135
$\backslash\mathrm{um@usv@b fith}$	104, 982, 1014	$\backslash\mathrm{um@usv@b fs fitpartial}$	123, 224, 445, 450, 1136
$\backslash\mathrm{um@usv@b fitLatin}$	59, 490, 492, 783, 975, 1007, 1048, 1052	$\backslash\mathrm{um@usv@b fs fLatin}$	68, 1098, 1114
$\backslash\mathrm{um@usv@b fitlatin}$	60, 495, 497, 784, 977, 1013, 1049, 1053	$\backslash\mathrm{um@usv@b fs flatin}$	69, 70, 1099, 1115
$\backslash\mathrm{um@usv@b fitNabla}$	115, 205, 439, 447, 995, 1057	$\backslash\mathrm{um@usv@b fs fNabla}$	116, 202, 440, 448
$\backslash\mathrm{um@usv@b fitpartial}$	121, 223, 443, 449, 996, 1040, 1058	$\backslash\mathrm{um@usv@b fs fnum}$	67
$\backslash\mathrm{um@usv@b fitvarepsilon}$	106, 508, 516, 997, 1034, 1059	$\backslash\mathrm{um@usv@b fs fpartial}$	122, 220, 444, 450
$\backslash\mathrm{um@usv@b fitvarkappa}$	108, 510, 518, 999, 1036, 1061	$\backslash\mathrm{um@usv@b fs fuplatin}$	70
$\backslash\mathrm{um@usv@b fitvarphi}$	109, 511, 519, 1000, 1037, 1062	$\backslash\mathrm{um@usv@b fuph}$	103, 1011
$\backslash\mathrm{um@usv@b fitvarpi}$	111, 513, 521, 1002, 1039, 1064	$\backslash\mathrm{um@usv@b fuplatin}$	56
$\backslash\mathrm{um@usv@b fitvarrho}$	110, 512, 520, 1001, 1038, 1063	$\backslash\mathrm{um@usv@b fvarepsilon}$	88, 508, 516, 987, 1025, 1079
$\backslash\mathrm{um@usv@b fitvarTheta}$	105, 501, 504, 994, 1021, 1056	$\backslash\mathrm{um@usv@b fvarkappa}$	90, 510, 518, 989, 1027, 1081
$\backslash\mathrm{um@usv@b fitvartheta}$	107, 509, 517, 998, 1035, 1060	$\backslash\mathrm{um@usv@b fvarphi}$	91, 511, 519, 990, 1028, 1082
$\backslash\mathrm{um@usv@b fLatin}$	54, 490, 492, 781, 974, 1005, 1052, 1068, 1072	$\backslash\mathrm{um@usv@b fvarpi}$	93, 513, 521, 992, 1030, 1084
$\backslash\mathrm{um@usv@b flatin}$	55, 56, 495, 497, 782, 976, 1010, 1053, 1069, 1073	$\backslash\mathrm{um@usv@b fvarrho}$	92, 512, 520, 991, 1029, 1083
$\backslash\mathrm{um@usv@b fNabla}$	114, 201, 438, 447, 984, 1077	$\backslash\mathrm{um@usv@b fvarTheta}$	86, 501, 504, 983, 1018, 1076
$\backslash\mathrm{um@usv@b fnum}$	53, 970, 1047, 1067, 1087, 1092, 1097, 1113, 1129	$\backslash\mathrm{um@usv@b fvartheta}$	89, 509, 517, 988, 1026, 1080
$\backslash\mathrm{um@usv@b fpartial}$	120, 219, 442, 449, 986, 1031, 1078	$\backslash\mathrm{um@usv@D igamma}$	78, 971, 985
$\backslash\mathrm{um@usv@b fscr Latin}$	65, 1093	$\backslash\mathrm{um@usv@digamma}$	85, 972, 993
		$\backslash\mathrm{um@usv@frak Latin}$	43, 946, 1088
		$\backslash\mathrm{um@usv@frak latin}$	44, 952, 1089
		$\backslash\mathrm{um@usv@itGreek}$	36, 777, 808, 811, 890, 906, 979, 1017, 1020, 1050, 1070, 1100, 1116, 1132
		$\backslash\mathrm{um@usv@itgreek}$	37, 817, 825, 891, 907, 981, 1024, 1033, 1051, 1071, 1101, 1117, 1133
		$\backslash\mathrm{um@usv@ith}$	95, 773, 800, 803, 905, 982, 1011, 1014

$\backslash\mathrm{um@usv@itLatin}$	32, 771, 792, 794, 888, 903, 920, 931, 946, 956, 961, 966, 975, 1005, 1007, 1048, 1068, 1088, 1093, 1098, 1114, 1130
$\backslash\mathrm{um@usv@itlatin}$	33, 772, 799, 802, 889, 904, 928, 940, 952, 957, 962, 967, 977, 1010, 1013, 1049, 1069, 1089, 1094, 1099, 1115, 1131
$\backslash\mathrm{um@usv@itNabla}$	113, 204, 430, 434, 892, 908, 995, 1042, 1057, 1077, 1103, 1119, 1135
$\backslash\mathrm{um@usv@itpartial}$. . .	119, 222, 432, 435, 893, 909, 996, 1031, 1040, 1043, 1058, 1078, 1104, 1120, 1136
$\backslash\mathrm{um@usv@itvarepsilon}$	97, 818, 826, 895, 911, 997, 1025, 1034, 1059, 1079, 1105, 1121, 1137
$\backslash\mathrm{um@usv@itvarkappa}$	99, 820, 828, 897, 913, 999, 1027, 1036, 1061, 1081, 1107, 1123, 1139
$\backslash\mathrm{um@usv@itvarphi}$	100, 821, 829, 898, 914, 1000, 1028, 1037, 1062, 1082, 1108, 1124, 1140
$\backslash\mathrm{um@usv@itvarpi}$	102, 823, 831, 900, 916, 1002, 1030, 1039, 1064, 1084, 1110, 1126, 1142
$\backslash\mathrm{um@usv@itvarrho}$	101, 822, 830, 899, 915, 1001, 1029, 1038, 1063, 1083, 1109, 1125, 1141
$\backslash\mathrm{um@usv@itvarTheta}$	96, 812, 894, 910, 994, 1018, 1021, 1056, 1076, 1102, 1118
$\backslash\mathrm{um@usv@itvartheta}$	98, 819, 827, 896, 912, 998, 1026, 1035, 1060, 1080, 1106, 1122, 1138
$\backslash\mathrm{um@usv@Nabla}$	112, 200, 429, 434, 892, 908, 984, 1042, 1057, 1077, 1103, 1119, 1135
$\backslash\mathrm{um@usv@num}$	29, 477, 919, 955, 960, 965, 970, 1047, 1067, 1087, 1092, 1097, 1113, 1129
$\backslash\mathrm{um@usv@partial}$	118, 218, 431, 435, 893, 909, 986, 1031, 1040, 1043, 1058, 1078, 1104, 1120, 1136
$\backslash\mathrm{um@usv@scrLatin}$	41, 931
$\backslash\mathrm{um@usv@scrlatin}$	42, 940
$\backslash\mathrm{um@usv@sfitLatin}$	48, 961
$\backslash\mathrm{um@usv@sfitlatin}$	49, 962
$\backslash\mathrm{um@usv@sflatin}$	46, 956
$\backslash\mathrm{um@usv@sflatin}$	47, 957
$\backslash\mathrm{um@usv@sfnun}$	45, 955, 960
$\backslash\mathrm{um@usv@ttLatin}$	51, 966
$\backslash\mathrm{um@usv@ttlatin}$	52, 967
$\backslash\mathrm{um@usv@ttnum}$	50, 965
$\backslash\mathrm{um@usv@upGreek}$	34, 775, 808, 811, 890, 906, 978, 1017, 1020, 1050, 1070, 1100, 1116, 1132
$\backslash\mathrm{um@usv@upgreek}$	35, 778, 817, 825, 891, 907, 980, 1024, 1033, 1051, 1071, 1101, 1117, 1133
$\backslash\mathrm{um@usv@upLatin}$	30, 770, 792, 794, 888, 903, 920, 931, 946, 956, 961, 966, 974, 1005, 1007, 1048, 1068, 1088, 1093, 1098, 1114, 1130
$\backslash\mathrm{um@usv@uplatin}$	31, 774, 799, 802, 889, 904, 928, 940, 952, 957, 962, 967, 976, 1010, 1013, 1049, 1069, 1089, 1094, 1099, 1115, 1131
$\backslash\mathrm{um@usv@varepsilon}$	79, 818, 826, 895, 911, 987, 1025, 1034, 1059, 1079, 1105, 1121, 1137
$\backslash\mathrm{um@usv@varkappa}$	81, 820, 828, 897, 913, 989, 1027, 1036, 1061, 1081, 1107, 1123, 1139
$\backslash\mathrm{um@usv@varphi}$	82, 821, 829, 898, 914, 990, 1028, 1037, 1062, 1082, 1108, 1124, 1140
$\backslash\mathrm{um@usv@varpi}$	84, 823, 831, 900, 916, 992, 1030, 1039, 1064, 1084, 1110, 1126, 1142
$\backslash\mathrm{um@usv@varrho}$	83, 822, 830, 899, 915, 991, 1029, 1038, 1063, 1083, 1109, 1125, 1141
$\backslash\mathrm{um@usv@varTheta}$	77, 776, 809, 812, 894, 910, 983, 1018, 1021, 1056, 1076, 1102, 1118, 1134
$\backslash\mathrm{um@usv@vartheta}$	80, 819, 827, 896, 912, 988, 1026, 1035, 1060, 1080, 1106, 1122, 1138
$\backslash\mathrm{um@zf@feature}$	569, 581, 584
$\backslash\mathrm{um_bfNabla_up_or_it_usv}$	201, 205, 447, 1042

```

\um_bfpartial_up_or_it_usv .....
..... 219, 223, 449, 1043
\um_bfsfNabla_up_or_it_usv .....
..... 202, 206, 448
\um_bfsfpartial_up_or_it_usv .....
..... 220, 224, 450
\um_config_mathbb: ..... 918
\um_config_mathbf: ..... 969
\um_config_mathbffrac: ..... 1086
\um_config_mathbfbit: ..... 1046
\um_config_mathbfscr: ..... 1091
\um_config_mathbfsf: ..... 1096
\um_config_mathbfsfit: ..... 1128
\um_config_mathbfsfup: ..... 1112
\um_config_mathbfup: ..... 1066
\um_config_mathfrac: ..... 945
\um_config_mathit: ..... 902
\um_config_mathscr: ..... 930
\um_config_mathsf: ..... 954
\um_config_mathsfitt: ..... 959
\um_config_mathtt: ..... 964
\um_config_mathup: ..... 887
\um_fix_mathtt: ..... 847
\um_glyph_if_exist:n ..... 853
\um_glyph_if_exist:nTF .....
..... 835, 853, 1178, 1181, 1184
\um_init_alphabet:n ..... 396, 850
\um_make_mathactive:nNn . 464–467, 469
\um_map_char:nn ..... 501, 504,
508–513, 516–521, 674, 773, 776,
800, 803, 809, 812, 818–823, 826–831
\um_map_char:nn_ ..... 655
\um_map_chars_greek:nn .....
... 500, 503, 507, 515, 668, 775,
777, 778, 785–788, 808, 811, 817, 825
\um_map_chars_latin:nn .....
490, 492, 495, 497, 665, 770–772,
774, 781–784, 792, 794, 799, 802
\um_map_chars_numbers:nn ... 477, 671
\um_map_chars_range:nnn .....
..... 655, 666, 669, 672, 675
\um_mathmap:Nnn ..... 394, 401, 680, 687
\um_mathmap_noparse:Nnn .... 394, 527
\um_mathmap_parse:Nnn ..... 401, 532
\um_maybe_init_alphabet:n 396, 403, 836
\um_Nabla_up_or_it_usv .. 200, 204, 434
\um_nprimes:n 1170, 1178, 1181, 1184, 1187
\um_nprimes_select:n ..... 1174, 1206
\um_partial_up_or_it_usv 218, 222, 435
\um_peek_execute_branches_ss: ...
..... 1340, 1345
\um_peek_execute_branches_ss_aux:
..... 1351, 1364
\um_prepare_alph:n ..... 837, 856
\um_process_symbol_noparse:nnnn .
..... 393, 418
\um_process_symbol_parse:nnnn 400, 418
\um_remap_symbol:nnn . 395, 402, 427,
429–432, 434, 435, 438–445, 447–450
\um_remap_symbol_noparse:nnn 395, 426
\um_remap_symbol_parse:nnn .....
..... 402, 426, 453
\um_remap_symbols: ..... 409, 426
\um_scan_ssript: 1254, 1287, 1325, 1327
\um_scan_ssript:TF ..... 1326, 1335
\um_scanprime: .....
..... 1190, 1200, 1212, 1216, 1217
\um_scanprime_collect: .....
..... 1193, 1195, 1198, 1201, 1204
\um_set_mathalph_range:Nnn ..... 684
\um_set_mathalph_range:nNnn ....
..... 684, 694, 697, 700
\um_set_mathalphabet_char:Nnn ...
..... 678, 892–900,
905, 908–916, 921–927, 932–939,
941–943, 947–951, 971, 972,
982–1002, 1011, 1014, 1018, 1021,
1025–1031, 1034–1040, 1042,
1043, 1056–1064, 1076–1084,
1102–1110, 1118–1126, 1134–1142
\um_set_mathalphabet_char:Nnnn .. 677
\um_set_mathalphabet_greek:Nnn ..
..... 699, 890, 891,
906, 907, 978–981, 1017, 1020,
1024, 1033, 1050, 1051, 1054,
1055, 1070, 1071, 1074, 1075,
1100, 1101, 1116, 1117, 1132, 1133
\um_set_mathalphabet_latin:Nnn ..
... 696, 888, 889, 903, 904, 920,
928, 931, 940, 946, 952, 956, 957,
961, 962, 966, 967, 974–977, 1005,
1007, 1010, 1013, 1048, 1049,

```

1052, 1053, 1068, 1069, 1072, 1073, 1088, 1089, 1093, 1094, 1098, 1099, 1114, 1115, 1130, 1131	\usepackage 3, 4, 6, 38, 39, 41
\um_set_mathalphabet_numbers: Nnn 693, 919, 955, 960, 965, 970, 1047, 1067, 1087, 1092, 1097, 1113, 1129	V
\um_set_mathcode: nnnn 347, 460, 544, 658	\varepsilon 758
\um_setup_active_subscript: nn 1278, 1292–1320	\varkappa 762
\um_setup_active_superscript: nn 1245, 1259–1275	\varointclockwise 549
\um_setup_alphabets: 412, 869	\varphi 763
\um_setup_alphanum: 411, 475	\varpi 767
\um_setup_bf_literals: 487, 780	\varrho 766
\um_setup_Greek: 483, 806	\varsigma 748
\um_setup_greek: 484, 815	\varTheta 721
\um_setup_Latin: 481, 790	\vartheta 761
\um_setup_latin: 482, 797	\vbox 24, 32, 71, 79
\um_setup_literals: 479, 769	\version@elt 250
\um_setup_math_alphabet: n 834, 870–885	\version@list 249
\um_setup_mathactives: 410, 463	\voffset 22, 69
\um_setup_mathup: 1378	W
\um_setup_nabla: 198, 415	\wd 31, 78
\um_setup_partial: 216, 416	X
\um_setup_shapes: 408, 414	\xdef 563, 596
\um_sub_or_super: n . . 1252, 1285, 1329	\XeTeXdelcode 329, 334
\um_symfont_tl 391, 399, 405, 419, 460, 471, 529, 536, 660	\XeTeXdelimiter 328, 333
\UnicodeMathSymbol 393, 400, 1162	\XeTeXmathaccent 338
\unless 600	\XeTeXmathchardef 312, 470
\updefault 406, 848	\XeTeXmathcode 330, 335, 340, 348
\upGreek 9, 44	\XeTeXmathcodenum 473, 1248, 1281
\upgreek 10, 45	\XeTeXradical 325
\upint 552	\Xi 717
\upLatin 7, 42	\xi 744
\uplatin 8, 43	\XKV@rm 376
\Upsilon 724	Z
\upsilon 751	\Z 927, 951
\use: c 838, 843, 859, 865	\z@ 1221, 1224, 1225, 1229
\use_none: n 403	\Zeta 709
	\zeta 736
	\zf@family 406
	\zf@fontspec 363
	\zf@update@ff 582, 585