

# Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/09/11      VO.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

<b>1</b>	<b>Introduction</b>	<b>2</b>	5.2	Overcoming \@on-lypreamble	16
<b>2</b>	<b>Specification</b>	<b>2</b>	5.3	Other things	16
	2.1 Using multiple fonts	3	<b>6</b>	<b>Fundamentals</b>	<b>17</b>
	2.2 Script and scriptscript fonts/features	3	6.1	Enlarging the number of maths families	17
<b>3</b>	<b>Maths input</b>	<b>3</b>	6.2	\DeclareMathSymbol for unicode ranges	17
	3.1 Miscellanea	4	6.3	The main \setmathfont macro	20
<b>4</b>	<b>Package options</b>	<b>4</b>	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 special attention	6	6.7	Maths accents	34
<b>I</b>	<b>The unicode-math package</b>	<b>9</b>	<b>7</b>	<b>Font features</b>	<b>35</b>
<b>5</b>	<b>Things we need</b>	<b>9</b>	7.1	OpenType maths font features	35
	5.1 Package options	13	7.2	Script and scriptscript font options	35
			7.3	Range processing	35

<b>8 Maths alphabets mapping definitions</b>	<b>43</b>	A.1 Overview	59
8.1 Bold alphabets' character mappings	48	<b>III X<sub>Y</sub>TeX math font dimensions</b>	<b>60</b>
8.2 Definitions of the math symbols	53	<b>IV Some manner of unit testing</b>	<b>65</b>
8.3 Epilogue	54	<b>B The regular weight alphabets</b>	<b>66</b>
<b>II stix table data extraction</b>	<b>58</b>	<b>C The bold alphabets</b>	<b>67</b>
<b>A Documenting maths support in the NFSS</b>	<b>59</b>		

## 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<sub>Y</sub>TeX, although it is conjectured that some effort could be spent to create a cross-format package that would also work with LuaTeX.

## 2 Specification

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

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

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

would implement this for every every symbol and alphabetic variant. That means `x` to  $x$ , `\xi` to  $\xi$ , `\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<sup>A</sup>TeX 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  $A_{B_C}$ ).

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

$\XTeX$ ’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

## 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`, `\primetriple`, 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
```

## 4 Package options

### 4.1 Math ‘style’

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

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

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

If glyphs are desired that do not map as per the package option (for example, an upright ‘ $g$ ’ is desired but typing `$g$` yields ‘ $g$ ’), *markup* is required to specify

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

Package option	Example	
	$(a, z, B, X)$	$(\alpha, \beta, \Gamma, \Xi)$
<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>	$(a, z, B, X)$	$(\alpha, \beta, \Gamma, \Xi)$

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. Table ?? on page ?? shows every character under the effect of this package option.

## 4.2 Bold switching

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

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

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

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

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

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

Package option	Example	
	$(a, z, B, X)$	$(\alpha, \beta, \Gamma, \Xi)$
<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	$\nabla$
	Bold serif	$\boldsymbol{\nabla}$
	Bold sans	$\nabla$
Italic	Serif	$\nabla$
	Bold serif	$\boldsymbol{\nabla}$
	Bold sans	$\nabla$

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. Table ?? on page ?? shows every character under the effect of this package option.

### 4.3 Symbols requiring special attention

**Nabla** The symbol  $\nabla$  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).  $\text{\TeX}$  classically uses an upright nabla, but ISO standards differ (I think). The package options `nabla=upright` and `nabla=italic` switch between the two choices. This is then inherited through `\mathbf{}`, `\mathit{}` and `\mathup{}` 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`.

**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.

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	$\partial$
	Italic	$\partial$
Bold	Upright	<b><math>\partial</math></b>
	Italic	<b><math>\partial</math></b>
Sans bold	Upright	$\partial$
	Italic	$\partial$

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`.<sup>1</sup>

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

**Epsilon and phi:  $\varepsilon$  vs.  $\epsilon$  and  $\varphi$  vs.  $\phi$**   $\text{\TeX}$  defines `\epsilonpsilon` to look like  $\epsilon$  and `\varepsilonpsilon` to look like  $\varepsilon$ . The Unicode glyph directly after delta and before zeta is ‘epsilon’ and looks like  $\varepsilon$ ; there is a subsequent variant of epsilon that looks like  $\epsilon$ . This creates a problem. People who use unicode input won’t want their glyphs transforming;  $\text{\TeX}$  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 `\epsilonpsilon` produce  $\phi$  and  $\varepsilon$  and `\varphi` and `\varepsilonpsilon` produce  $\phi$  and  $\epsilon$ . 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

**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:

<sup>1</sup>A 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.

$A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$   
 $A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$

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

$A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$   
 $A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$

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

$A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$   
 $A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$

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

Figure 1: Example maths output demonstrating the math-style package option.

$A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$   
 $A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$

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

$A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$   
 $A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$

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

$A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$   
 $A B C D E F G H I J K L M N O P Q R S T U V W X Y Z$   
 $a b c d e f g h i j k l m n o p q r s t u v w x y z$

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

Figure 2: Example maths output demonstrating the bold-style package option.



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

## File I

# The unicode-math package

This is the package.

```
1 \ProvidesPackage{unicode-math}
2 [2009/09/11 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<sup>A</sup>T<sub>E</sub>X<sub>3</sub> — finally!

```
6 \ExplSyntaxOn
```

### Counters and conditionals

```
7 \newcounter{um@fam}
8 \newif\if@um@fontspec@feature
9 \newif\if@um@ot@math@
```

For math-style:

```
10 \newif\if@um@literal
11 \newif\if@um@upGreek
12 \newif\if@um@upgreek
13 \newif\if@um@upLatin
14 \newif\if@um@uplatin
```

For bold-style:

```
15 \newif\if@um@bfliteral
16 \newif\if@um@bfupGreek
17 \newif\if@um@bfupgreek
18 \newif\if@um@bfupLatin
19 \newif\if@um@bfuplatin
```

For nabla:

```
20 \newif\if@um@upNabla
21 \newif\if@um@uppartial
22 \bool_new:N \g_um_texgreek_bool
```

## Programming niceties

```
\um@Loop See Kees van der Laan's various articles on TEX programming:
\um@Break
23 \def\um@Loop#1\um@Pool{#1\um@Loop#1\um@Pool}
24 \def\um@Break#1\um@Pool{}
```

## Shortcuts

```
25 \newcommand\um@PackageError[2]{\PackageError{unicode-math}{#1}{#2}}
26 \newcommand\um@PackageWarning[1]{\PackageWarning{unicode-math}{#1}}
27 \newcommand\um@PackageInfo[1]{\PackageInfo{unicode-math}{#1}}
```

### 5.0.1 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.<sup>2</sup>

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

---

<sup>2</sup>'u.s.v.' stands for 'unicode scalar value'.

Bold:

```
52 \def\um@usv@bfnum{"1D7CE}
53 \def\um@usv@bfLatin{"1D400}
54 \def\um@usv@bflatin{"1D41A}
55 \let\um@usv@bfuplatin\um@usv@bflatin
56 \def\um@usv@bfGreek{"1D6A8}
57 \def\um@usv@bfgreek{"1D6C2}
58 \def\um@usv@bfitLatin{"1D468}
59 \def\um@usv@bfitlatin{"1D482}
60 \def\um@usv@bfitGreek{"1D71C}
61 \def\um@usv@bfitgreek{"1D736}
62 \def\um@usv@bffrakLatin{"1D56C}
63 \def\um@usv@bffraklatin{"1D586}
64 \def\um@usv@bfscrLatin{"1D4D0}
65 \def\um@usv@bfscrlatin{"1D4EA}
66 \def\um@usv@bfsfnum{"1D7EC}
67 \def\um@usv@bfsfLatin{"1D5D4}
68 \def\um@usv@bfsflatin{"1D5EE}
69 \let\um@usv@bfsfuplatin\um@usv@bfsflatin
70 \def\um@usv@bfsfGreek{"1D756}
71 \def\um@usv@bfsfgreek{"1D770}
72 \def\um@usv@bfsfitLatin{"1D63C}
73 \def\um@usv@bfsfitlatin{"1D656}
74 \def\um@usv@bfsfitGreek{"1D790}
75 \def\um@usv@bfsfitgreek{"1D7AA}
```

Greek variants:

```
76 \def\um@usv@varTheta{"3F4}
77 \def\um@usv@Digamma{"3DC}
78 \def\um@usv@varepsilon{"3F5}
79 \def\um@usv@vartheta{"3D1}
80 \def\um@usv@varkappa{"3F0}
81 \def\um@usv@varphi{"3D5}
82 \def\um@usv@varrho{"3F1}
83 \def\um@usv@varpi{"3D6}
84 \def\um@usv@digamma{"3DD}
85 \tl_new:Nn \g_um_up_epsilon_letter_usv {"25B}
86 \tl_new:Nn \g_um_up_epsilon_symbol_usv {"3F5}
87 \tl_new:Nn \g_um_up_phi_letter_usv {"3C6}
88 \tl_new:Nn \g_um_up_phi_symbol_usv {"3D5}
```

Bold:

```
89 \def\um@usv@bfvarTheta{"1D6B9}
90 \def\um@usv@bfDigamma{"1D7CA}
91 \def\um@usv@bfvarepsilon{"1D6DC}
92 \def\um@usv@bfvartheta{"1D6DD}
93 \def\um@usv@bfvarkappa{"1D6DE}
```

```

94 \def\um@usv@bfvarphi{"1D6DF}
95 \def\um@usv@bfvarrho{"1D6E0}
96 \def\um@usv@bfvarpi{"1D6E1}
97 \def\um@usv@bfdigamma{"1D7CB}
98 \tl_new:Nn \g_um_bfup_epsilon_letter_usv {"1D6C6}
99 \tl_new:Nn \g_um_bfup_epsilon_symbol_usv {"1D6DC}
100 \tl_new:Nn \g_um_bfup_phi_letter_usv {"1D6D7}
101 \tl_new:Nn \g_um_bfup_phi_symbol_usv {"1D6DF}

```

Italic Greek variants:

```

102 \def\um@usv@ith{"210E}
103 \def\um@usv@itvarTheta{"1D6F3}
104 \def\um@usv@itvarepsilon{"1D716}
105 \def\um@usv@itvartheta{"1D717}
106 \def\um@usv@itvarkappa{"1D718}
107 \def\um@usv@itvarphi{"1D719}
108 \def\um@usv@itvarrho{"1D71A}
109 \def\um@usv@itvarpi{"1D71B}
110 \tl_new:Nn \g_um_it_epsilon_symbol_usv {"1D716}
111 \tl_new:Nn \g_um_it_epsilon_letter_usv {"1D700}
112 \tl_new:Nn \g_um_it_phi_symbol_usv {"1D719}
113 \tl_new:Nn \g_um_it_phi_letter_usv {"1D711}

```

Bold:

```

114 \def\um@usv@bfuph{"1D421}
115 \def\um@usv@bfith{"1D489}
116 \def\um@usv@bfitvarTheta{"1D72D}
117 \def\um@usv@bfitvarepsilon{"1D750}
118 \def\um@usv@bfitvartheta{"1D751}
119 \def\um@usv@bfitvarkappa{"1D752}
120 \def\um@usv@bfitvarphi{"1D753}
121 \def\um@usv@bfitvarrho{"1D754}
122 \def\um@usv@bfitvarpi{"1D755}
123 \tl_new:Nn \g_um_bfit_epsilon_letter_usv {"1D73A}
124 \tl_new:Nn \g_um_bfit_epsilon_symbol_usv {"1D750}
125 \tl_new:Nn \g_um_bfit_phi_letter_usv {"1D74B}
126 \tl_new:Nn \g_um_bfit_phi_symbol_usv {"1D753}

```

Nabla:

```

127 \def\um@usv@Nabla{"2207}
128 \def\um@usv@itNabla{"1D6FB}
129 \def\um@usv@bfNabla{"1D6C1}
130 \def\um@usv@bfitNabla{"1D735}
131 \def\um@usv@bfsfNabla{"1D76F}
132 \def\um@usv@bfsfitNabla{"1D7A9}

```

Partial:

```

133 \def\um@usv@partial{"2202}

```

```

134 \def\um@usv@itpartial{"1D715}
135 \def\um@usv@bfpartial{"1D6DB}
136 \def\um@usv@bfitpartial{"1D74F}
137 \def\um@usv@bfsfpartial{"1D789}
138 \def\um@usv@bfsfitpartial{"1D7C3}

```

## 5.1 Package options

xkeyval's package support is used here.

### math-style

```

139 \define@choicekey*{unicode-math.sty}
140   {math-style}[\@tempa\@tempb]{iso,tex,french,literal}{
141   \ifcase\@tempb\relax
142     \um@upGreekfalse
143     \um@upgreekfalse
144     \um@upLatinfalse
145     \um@uplatinfalse
146     \um@bfupGreekfalse
147     \um@bfupgreekfalse
148     \um@uppartialfalse
149     \um@bfupLatinfalse
150     \um@bfuplatinfalse
151     \um@upNablafalse
152     \bool_set_false:N \g_um_texgreek_bool
153   \or
154     \um@upGreektrue
155     \um@upgreekfalse
156     \um@upLatinfalse
157     \um@uplatinfalse
158     \um@bfupGreektrue
159     \um@bfupgreekfalse
160     \um@uppartialfalse
161     \um@bfupLatintrue
162     \um@bfuplatintrue
163     \um@upNabltrue
164     \bool_set_true:N \g_um_texgreek_bool
165   \or
166     \um@upGreektrue
167     \um@upgreektrue
168     \um@upLatintrue
169     \um@uplatinfalse
170     \um@bfupGreektrue
171     \um@bfupgreektrue
172     \um@uppartialtrue
173     \um@bfupLatintrue

```

```

174 \um@bfuplatintrue
175 \um@upNablatrue
176 \bool_set_false:N \g_um_texgreek_bool
177 \or
178 \um@literaltrue
179 \um@bfliteraltrue
180 \bool_set_false:N \g_um_texgreek_bool
181 \fi
182 }

```

### bold-style

```

183 \define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,french,literal}{
184 \ifcase\@tempb\relax
185 \um@bfupGreekfalse
186 \um@bfupgreekfalse
187 \um@uppartialfalse
188 \um@bfupLatinfalse
189 \um@bfuplatinfalse
190 \or
191 \um@bfupGreektrue
192 \um@bfupgreekfalse
193 \um@uppartialfalse
194 \um@bfupLatintrue
195 \um@bfuplatintrue
196 \or
197 \um@bfupGreektrue
198 \um@bfupgreektrue
199 \um@uppartialtrue
200 \um@bfupLatintrue
201 \um@bfuplatintrue
202 \or
203 \um@bfliteraltrue
204 \fi
205 }

```

### Symbol obliqueness

```

206 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
207 \ifcase\@tempb\relax
208 \um@upNablatrue
209 \or
210 \um@upNablafalse
211 \fi
212 }
213 \cs_set:Nn \um_setup_nabla: {
214 \if@um@upNabla

```

```

215 \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@Nabla }
216 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfNabla }
217 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfNabla }
218 \else
219 \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@itNabla }
220 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfitNabla }
221 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfitNabla }
222 \fi
223 }
224 \define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
225 \ifcase\@tempb\relax
226 \um@uppartialtrue
227 \or
228 \um@uppartialfalse
229 \fi
230 }
231 \cs_set:Nn \um_setup_partial: {
232 \ifum@uppartial
233 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@partial }
234 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfpartial }
235 \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfpartial }
236 \else
237 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@itpartial }
238 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfitpartial }
239 \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfitpartial }
240 \fi
241 }

```

### Epsilon and phi shapes

```

242 \define@choicekey*{unicode-math.sty}{vargreek-shape}[\@tempa\@tempb]{unicode,TeX}{
243 \ifcase\@tempb\relax
244 \bool_set_false:N \g_um_texgreek_bool
245 \or
246 \bool_set_true:N \g_um_texgreek_bool
247 \fi
248 }
249 \cs_set:Nn \um_setup_vargreek: {
250 \bool_if:NTF \g_um_texgreek_bool {
251 \num_set_eq:NN \g_um_up_phi_usv \g_um_up_phi_symbol_usv
252 \num_set_eq:NN \g_um_up_varphi_usv \g_um_up_phi_letter_usv
253 \num_set_eq:NN \g_um_it_phi_usv \g_um_it_phi_symbol_usv
254 \num_set_eq:NN \g_um_it_varphi_usv \g_um_it_phi_letter_usv
255 \num_set_eq:NN \g_um_bfup_phi_usv \g_um_bfup_phi_symbol_usv
256 \num_set_eq:NN \g_um_bfup_varphi_usv \g_um_bfup_phi_letter_usv
257 \num_set_eq:NN \g_um_bfit_phi_usv \g_um_bfit_phi_symbol_usv
258 \num_set_eq:NN \g_um_bfit_varphi_usv \g_um_bfit_phi_letter_usv

```

```

259 \num_set_eq:NN \g_um_up_epsilon_usv \g_um_up_epsilon_symbol_usv
260 \num_set_eq:NN \g_um_up_varepsilon_usv \g_um_up_epsilon_letter_usv
261 \num_set_eq:NN \g_um_it_epsilon_usv \g_um_it_epsilon_symbol_usv
262 \num_set_eq:NN \g_um_it_varepsilon_usv \g_um_it_epsilon_letter_usv
263 \num_set_eq:NN \g_um_bfup_epsilon_usv \g_um_bfup_epsilon_symbol_usv
264 \num_set_eq:NN \g_um_bfup_varepsilon_usv \g_um_bfup_epsilon_letter_usv
265 \num_set_eq:NN \g_um_bfit_epsilon_usv \g_um_bfit_epsilon_symbol_usv
266 \num_set_eq:NN \g_um_bfit_varepsilon_usv \g_um_bfit_epsilon_letter_usv
267 }{
268 \num_set_eq:NN \g_um_up_varphi_usv \g_um_up_phi_symbol_usv
269 \num_set_eq:NN \g_um_up_phi_usv \g_um_up_phi_letter_usv
270 \num_set_eq:NN \g_um_it_varphi_usv \g_um_it_phi_symbol_usv
271 \num_set_eq:NN \g_um_it_phi_usv \g_um_it_phi_letter_usv
272 \num_set_eq:NN \g_um_bfup_varphi_usv \g_um_bfup_phi_symbol_usv
273 \num_set_eq:NN \g_um_bfup_phi_usv \g_um_bfup_phi_letter_usv
274 \num_set_eq:NN \g_um_bfit_varphi_usv \g_um_bfit_phi_symbol_usv
275 \num_set_eq:NN \g_um_bfit_phi_usv \g_um_bfit_phi_letter_usv
276 \num_set_eq:NN \g_um_up_varepsilon_usv \g_um_up_epsilon_symbol_usv
277 \num_set_eq:NN \g_um_up_epsilon_usv \g_um_up_epsilon_letter_usv
278 \num_set_eq:NN \g_um_it_varepsilon_usv \g_um_it_epsilon_symbol_usv
279 \num_set_eq:NN \g_um_it_epsilon_usv \g_um_it_epsilon_letter_usv
280 \num_set_eq:NN \g_um_bfup_varepsilon_usv \g_um_bfup_epsilon_symbol_usv
281 \num_set_eq:NN \g_um_bfup_epsilon_usv \g_um_bfup_epsilon_letter_usv
282 \num_set_eq:NN \g_um_bfit_varepsilon_usv \g_um_bfit_epsilon_symbol_usv
283 \num_set_eq:NN \g_um_bfit_epsilon_usv \g_um_bfit_epsilon_letter_usv
284 }
285 }
286 \ExecuteOptionsX{math-style=TeX}
287 \ProcessOptionsX

```

## 5.2 Overcoming \@onlypreamble

This will be refined later! Sort out which macros actually have to be removed from the \@preamblecmds token list. There is a macro to remove items from the \@preamblecmds list in gmutils.sty.

```
288 \def\@preamblecmds{}
```

## 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>

---

```

289 \def\um@fontdimen@percent#1#2{
290   0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax
291 }

```

`\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.

```

292 \def\um@scaled@apply#1#2#3{
293   \ifx#1\scriptstyle
294     #2\um@fontdimen@percent{10}\um@font#3
295   \else
296     \ifx#1\scriptscriptstyle
297       #2\um@fontdimen@percent{11}\um@font#3
298     \else
299       #2#3%
300     \fi
301   \fi
302 }

```

## 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 `ltxssbas.dtx`) we want to redefine

```

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

```

This is sufficient for L<sup>A</sup>T<sub>E</sub>X'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

```
305 \def \um@mathsymbol#1#2#3#4{
306   \expandafter\um@set@mathsymbol\csname sym#3\endcsname#1#2{#4}}
```

The final macros that actually define the maths symbol with  $\LaTeX$  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.

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

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

```
308 \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. `\unicodemathgobble` is the same as `\gobble` but needs to not have @ in its name because the argument goes inside a `\scantokens`.

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

```
309   \begingroup
310     \char_make_active:n {#4}
311     \global\mathcode#4="8000\relax
312     \um@scanactivedef #4 \@nil { \csname\string#2@op\endcsname }
313   \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`.

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

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

```
317   \cs_gset:cpn { \string#2 @op } {
318     \csname\string#2@sym\endcsname
319     \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
320     \ifin@
321       \expandafter\nolimits
322     \fi
323   }
```

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

```
324 \else
```

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

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

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

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

**Accents**

```
339 \ifx\mathaccent#3\relax
340 \cs_gset:Npx #2 {\XeTeXmathaccent "\mathchar@type#3 #1 #4\relax}
341 \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.

```
342 \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
343 \fi
344 \fi
345 \fi
346 \fi
347 \fi
348 }
```

`\SetMathCode` [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.

```
349 \newcommand\SetMathCode[4]{
350 \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
351 }
```

---

A

```
\zf@fontspec{}{Cambria Math}
\let\glb@currsizel\relax
\DeclareSymbolFont{test2}{EU1}{\zf@family}{m}{n}
\SetMathCode{65}{\mathalpha}{test2}{119860}
$A$
```

---

### 6.3 The main `\setmathfont` macro

Here's the simplest usage:

---

$Ax \stackrel{\text{def}}{=} \nabla \times Z$

```
\setmathfont{Cambria Math}
$Ax \eqdef \nabla \times \mscrZ$
```

---

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

---

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

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

---

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

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

- Erase any conception  $\text{\LaTeX}$  has of previously defined math symbol fonts; this allows `\DeclareSymbolFont` at any point in the document.

```
353 \let\glb@currsizel\relax
```

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

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

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

- Tell `fontspec` that maths font features are actually allowed.

```
356 \um@fontspec@featuretrue
```

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

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

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

```
358 \def\um@mversion{normal}
359 \DeclareMathVersion{\um@mversion}
```

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

```
360 \tl_set:Nn \l_um_script_features_tl {ScriptStyle}
361 \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
362 \tl_set:Nn \l_um_script_font_tl {#2}
363 \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.

```
364 \setkeys*[um]{options}{#1}
365 \edef\@tempa{\noexpand\zf@fontspec{
366   Script = Math,
367   SizeFeatures = {
368     {Size = \tf@size-} ,
369     {Size = \sf@size-\tf@size ,
370       Font = \l_um_script_font_tl ,
371         \l_um_script_features_tl
372     } ,
373     {Size = -\sf@size ,
374       Font = \l_um_sscript_font_tl ,
375         \l_um_sscript_features_tl
376     }
377   },
378   \XKV@rm
379 }{#2}
380 }
381 \@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`:

```
382 \font\um@font="#2"\relax
383 \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
384   \um@ot@math@true
385 \else
```

```

386 \um@PackageWarning{
387   The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
388   Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
389   in~ a~ substandard~ manner.
390 }
391 \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

```

392 \ifx\um@char@range\@empty
393   \def\um@symfont{um@allsym}
394   \um@PackageInfo{Defining~ the~ default~ maths~ font~ as~ '#2'}
395   \let \UnicodeMathSymbol \um@mathsymbol@noparse
396   \let \um_mathmap:Nnn \um_mathmap_noparse:Nnn
397   \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
398   \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
399 \else
400   \stepcounter{um@fam}
401   \edef\um@symfont{um@fam\theum@fam}
402   \let \UnicodeMathSymbol \um@mathsymbol@parse
403   \let \um_mathmap:Nnn \um_mathmap_parse:Nnn
404   \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
405   \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
406 \fi

```

Now defined `\um@symfont` as the L<sup>A</sup>T<sub>E</sub>X math font to access everything:

```

407 \DeclareSymbolFont{\um@symfont}
408   {\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`.

```

409 \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.)

```

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

End of the \setmathfont macro.
415 }

416 \cs_new:Nn \um_setup_shapes: {
417   \um_setup_nabla:
418   \um_setup_partial:
419   \um_setup_vargreek:
420 }

```

### 6.3.1 Functions for setting up symbols with mathcodes

`\um@mathsymbol@noparse`

```

421 \newcommand\um@mathsymbol@noparse[4]{
422   \um@mathsymbol{#2}{#3}{\um@symfont}{#1}
423 }

```

`\um@mathsymbol@parse` 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.

```

424 \newcommand\um@mathsymbol@parse[4]{
425   \um@parse@term{#1}{#2}{#3}{
426     %\um@PackageInfo{Defining \string#2 as mathchar #1}
427     \um@mathsymbol{#2}{#3}{\um@symfont}{#1}
428   }
429 }

```

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

```

430 \cs_new:Nn \um_remap_symbols: {
431   \um_remap_symbol:nnn{"2D}{\mathbin}{"02212}% hyphen to minus
432   \if@um@literal
433     \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
434     \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
435     \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
436     \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
437   \else
438     \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um@Nabla_up_or_it_usv}
439     \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um@partial_up_or_it_usv}
440   \fi

```

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

```

441   \if@um@bfliteral

```

```

442 \um_remap_symbol:nnn {\um@usv@bfNabla} {\mathord}{\um@usv@bfNabla}
443 \um_remap_symbol:nnn {\um@usv@bfitNabla} {\mathord}{\um@usv@bfitNabla}
444 \um_remap_symbol:nnn {\um@usv@bfsfNabla} {\mathord}{\um@usv@bfsfNabla}
445 \um_remap_symbol:nnn {\um@usv@bfsfitNabla} {\mathord}{\um@usv@bfsfitNabla}
446 \um_remap_symbol:nnn {\um@usv@bfpartial} {\mathord}{\um@usv@bfpartial}
447 \um_remap_symbol:nnn {\um@usv@bfitpartial} {\mathord}{\um@usv@bfitpartial}
448 \um_remap_symbol:nnn {\um@usv@bfsfpartial} {\mathord}{\um@usv@bfsfpartial}
449 \um_remap_symbol:nnn {\um@usv@bfsfitpartial} {\mathord}{\um@usv@bfsfitpartial}
450 \else
451 \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla} {\mathord}{\um@usv@bfNabla_up_or_it_usv}
452 \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla} {\mathord}{\um@usv@bfsfNabla_up_or_it_usv}
453 \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial} {\mathord}{\um@usv@bfpartial_up_or_it_usv}
454 \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial} {\mathord}{\um@usv@bfsfpartial_up_or_it_usv}
455 \fi
456 }

```

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

```

457 \cs_new:Nn \um_remap_symbol_parse:nnn {
458   \um@parse@term {#3} {\@nil} {#2} {
459     \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
460   }
461 }
462 \cs_new:Nn \um_remap_symbol_noparse:nnn {
463   \clist_map_inline:nn {#1} {
464     \SetMathCode {##1} {#2} {\um@symfont} {#3}
465   }
466 }

```

### 6.3.2 Active math characters

`\um_setup_mathactives:`

```

467 \cs_new:Nn \um_setup_mathactives: {
468   \um_make_mathactive:nNN {"2032} \primesingle \mathord
469 }

```

`\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!

```

470 \cs_new:Nn \um_make_mathactive:nNN {
471   \XeTeXmathchardef #2 = "\mathchar@type #3
472     \csname sym\um@symfont\endcsname
473     #1 \scan_stop:
474   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:
475 }

```



### 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`.

```
476 \cs_set:Nn \um_setup_alphanum: {  
477   \ifx\um@char@range\@empty  
478     \um@def@numbers
```

#### Normal weight

```
479   \if@um@literal  
480     \um_setup_literals:  
481   \else  
482     \if@um@upLatin\um@def@upLatin\else\um@def@itLatin\fi  
483     \if@um@uplatin\um@def@uplatin\else\um@def@itlatin\fi  
484     \if@um@upGreek\um@def@upGreek\else\um@def@itGreek\fi  
485     \if@um@upgreek\um@def@upgreek\else\um@def@itgreek\fi  
486   \fi
```

#### Bold

```
487   \if@um@bfliteral  
488     \um_setup_bf_literals:  
489   \else  
490     \if@um@bfupLatin  
491       \um@setmathcode[26]{\um@usv@bfLatin,\um@usv@bfitLatin}{\um@usv@bfLatin}  
492     \else  
493       \um@setmathcode[26]{\um@usv@bfLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}  
494     \fi  
495     \if@um@bfuplatin  
496       \um@setmathcode[26]{\um@usv@bflatin,\um@usv@bfitlatin}{\um@usv@bflatin}  
497     \else  
498       \um@setmathcode[26]{\um@usv@bflatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}  
499     \fi  
500     \if@um@bfupGreek  
501       \um@setmathcode[25]{\um@usv@bfGreek,\um@usv@bfitGreek}{\um@usv@bfGreek}  
502       \um@setmathcode{\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}  
503     \else  
504       \um@setmathcode[25]{\um@usv@bfGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}  
505       \um@setmathcode{\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}  
506     \fi  
507     \if@um@bfupgreek  
508       \um@setmathcode[25]{\um@usv@bfgreek,\um@usv@bfitgreek}{\um@usv@bfgreek}  
509       \um@setmathcode{\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
```

```

510 \um@setmathcode{\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
511 \um@setmathcode{\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
512 \um@setmathcode{\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
513 \um@setmathcode{\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
514 \um@setmathcode{\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
515 \else
516 \um@setmathcode[25]{\um@usv@bfgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
517 \um@setmathcode{\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
518 \um@setmathcode{\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
519 \um@setmathcode{\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
520 \um@setmathcode{\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
521 \um@setmathcode{\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
522 \um@setmathcode{\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
523 \fi
524 \fi
525 \else
: TODO : what is supposed to happen here?
526 \fi
527 }

```

### 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 `\SetMathCode` 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.

```

528 \cs_set:Nn \um_mathmap_noparse:Nnn {
529   \clist_map_inline:nn {#2} {
530     \exp_args:No \um@addto@mathmap \um@symfont {##1}{#1}{#3}
531   }
532 }

```

`\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 `\SetMathCode` declarations to the maths alphabet definition (*e.g.*, `\um@mathscr`).

```

533 \cs_set:Nn \um_mathmap_parse:Nnn {
534   \clist_map_inline:Nn \um@char@num@range {
535     \ifnum##1=#3\relax
536     \clist_map_inline:nn {#2} {

```

```

537         \exp_args:No \um@addto@mathmap \um@symfont {####1}{#1}{#3}
538     }
539     \fi
540 }
541 }

```

`\um@addto@mathmap` #1 : Math symbol font, always/usually the expansion of `\um@symfont`  
#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` can be expanded before entering the `\g@addto@macro` command.

```

542 \newcommand\um@addto@mathmap[4]{
543   \expandafter\g@addto@macro
544   \csname um_setup_\cs_to_str:N #3:\endcsname{
545     \SetMathCode{#2}{\mathalpha}{#1}{#4}
546   }
547 }

```

























## 6.4 (Big) operators


























Turns out that  $\text{\XeTeX}$  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  $\text{\TeX}$  *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-maths.tex`, from which a subset need to be flagged for `\nolimits` adjustments. The limits behaviour as specified by `unicode-math` are shown (with grey ‘scripts’).

USV	Ex.	Macro	Description
U+02140	$\sum\limits_0^1$	<code>\Bbbsum</code>	DOUBLE-STRUCK N-ARY SUMMATION
U+0220F	$\prod\limits_0^1$	<code>\prod</code>	PRODUCT OPERATOR
U+02210	$\coprod\limits_0^1$	<code>\coprod</code>	COPRODUCT OPERATOR
U+02211	$\sum\limits_0^1$	<code>\sum</code>	SUMMATION OPERATOR
U+0222B	$\int\limits_0^1$	<code>\int</code>	INTEGRAL OPERATOR
U+0222C	$\iint\limits_0^1$	<code>\iint</code>	DOUBLE INTEGRAL OPERATOR
U+0222D	$\iiint\limits_0^1$	<code>\iiint</code>	TRIPLE INTEGRAL OPERATOR

U+0222E		<code>\oint</code>	CONTOUR INTEGRAL OPERATOR
U+0222F		<code>\oiint</code>	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230		<code>\oiint</code>	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231		<code>\intclockwise</code>	CLOCKWISE INTEGRAL
U+02232		<code>\varointclockwise</code>	CONTOUR INTEGRAL, CLOCKWISE
U+02233		<code>\ointctrackwise</code>	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0		<code>\bigwedge</code>	LOGICAL OR OPERATOR
U+022C1		<code>\bigvee</code>	LOGICAL AND OPERATOR
U+022C2		<code>\bigcap</code>	INTERSECTION OPERATOR
U+022C3		<code>\bigcup</code>	UNION OPERATOR
U+027D5		<code>\leftouterjoin</code>	LEFT OUTER JOIN
U+027D6		<code>\rightouterjoin</code>	RIGHT OUTER JOIN
U+027D7		<code>\fullouterjoin</code>	FULL OUTER JOIN
U+027D8		<code>\biguparrow</code>	LARGE UP TACK
U+027D9		<code>\bigdownarrow</code>	LARGE DOWN TACK
U+029F8		<code>\xsolidus</code>	BIG SOLIDUS
U+029F9		<code>\xbsolidus</code>	BIG REVERSE SOLIDUS
U+02A00		<code>\bigodot</code>	N-ARY CIRCLED DOT OPERATOR
U+02A01		<code>\bigoplus</code>	N-ARY CIRCLED PLUS OPERATOR
U+02A02		<code>\bigotimes</code>	N-ARY CIRCLED TIMES OPERATOR
U+02A03		<code>\bigcupdot</code>	N-ARY UNION OPERATOR WITH DOT
U+02A04		<code>\bigcupplus</code>	N-ARY UNION OPERATOR WITH PLUS
U+02A05		<code>\bigsqcap</code>	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06		<code>\bigsqcup</code>	N-ARY SQUARE UNION OPERATOR

U+02A07		\conjquant	TWO LOGICAL AND OPERATOR
U+02A08		\disjquant	TWO LOGICAL OR OPERATOR
U+02A09		\bigtimes	N-ARY TIMES OPERATOR
U+02A0B		\sumint	SUMMATION WITH INTEGRAL
U+02A0C		\iiint	QUADRUPLE INTEGRAL OPERATOR
U+02A0D		\intbar	FINITE PART INTEGRAL
U+02A0E		\intBar	INTEGRAL WITH DOUBLE STROKE
U+02A0F		\fint	INTEGRAL AVERAGE WITH SLASH
U+02A10		\cirfnint	CIRCULATION FUNCTION
U+02A11		\awint	ANTICLOCKWISE INTEGRATION LINE INTEGRATION WITH RECTANGULAR
U+02A12		\rppolint	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13		\scpolint	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE POLE
U+02A14		\npolint	INTEGRAL AROUND A POINT OPERATOR
U+02A15		\pointint	QUATERNION INTEGRAL OPERATOR
U+02A16		\sqint	INTEGRAL WITH LEFTWARDS ARROW
U+02A17		\intlarhk	WITH HOOK
U+02A18		\intx	INTEGRAL WITH TIMES SIGN
U+02A19		\intcap	INTEGRAL WITH INTERSECTION
U+02A1A		\intcup	INTEGRAL WITH UNION
U+02A1B		\upint	INTEGRAL WITH OVERBAR
U+02A1C		\lowint	INTEGRAL WITH UNDERBAR
U+02A1D		\Join	JOIN
U+02A1E		\bigtriangleleft	LARGE LEFT TRIANGLE OPERATOR
U+02A1F		\zcmp	Z NOTATION SCHEMA COMPOSITION
U+02A20		\zpipe	Z NOTATION SCHEMA PIPING
U+02A21		\zproject	Z NOTATION SCHEMA PROJECTION
U+02AFC		\biginterleave	LARGE TRIPLE VERTICAL BAR OPERATOR
U+02AFF		\bigtalloblong	N-ARY WHITE VERTICAL BAR

\um@nolimits This macro is a sequence containing those maths operators that require a \nolimits

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

```

548 \def\um@nolimits{
549   \@elt\int\@elt\iint\@elt\iiint\@elt\iiint\@elt\oint\@elt\oint\@elt\oiint
550   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
551   \@elt\intbar\@elt\intBar\@elt\oint\@elt\cirfnint\@elt\awint\@elt\rppoint
552   \@elt\scpolint\@elt\ntpolint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
553   \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
554 }

```

`\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`.

```

555 \newcommand\addnolimits[1]{
556   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
557 }

```

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

```

558 \def\removenolimits#1{
559   \begingroup
560     \def\@elt##1{
561       \ifx##1#1\else
562         \noexpand\@elt\noexpand##1
563       \fi}
564     \xdef\um@nolimits{\um@nolimits}
565   \endgroup
566 }

```

 $\iiint_V$ 
 $\iiint_V$ 
 $\iiint_V$ 

```

\def\dmath#1{\displaystyle #1}
\setmathfont{Cambria Math} \dmath{\iint_V}
\removenolimits\iint
\setmathfont{Cambria Math} \dmath{\iint_V}
\addnolimits\iint
\setmathfont{Cambria Math} \dmath{\iint_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.

567 `\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>

568 `\let\left@primitive\left`

569 `\def\left{\mathopen{}\left@primitive}`

No re-definition is made for `\right` because I don't believe it to be necessary.  
: TODO : 'fences', e.g., `\vert`

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

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

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

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

Here are all `\mathopen` characters:

USV	Ex.	Macro	Description
-----	-----	-------	-------------

U+00028	(	\lparen	LEFT PARENTHESIS
U+0005B	[	\lbrack	LEFT SQUARE BRACKET
U+0007B	{	\lbrace	LEFT CURLY BRACKET DOUBLE ANGLE QUOTATION MARK
U+000AB	«	\guillemotleft	(GUILLEMET), LEFT
U+02018	‘	\lq	SINGLE QUOTATION MARK, LEFT
U+0201A	,	\quotsinglbase	RISING SINGLE QUOTE, LEFT (LOW)
U+0201E	„	\quotdblbase	RISING DOUBLE QUOTE, LEFT (LOW) SINGLE ANGLE QUOTATION MARK
U+02039	<	\guilsinglleft	(GUILLEMET), LEFT
U+0221A	√	\sqrt	RADICAL
U+0221B	√ <sup>3</sup>	\cuberoot	CUBE ROOT
U+0221C	√ <sup>4</sup>	\fourthroot	FOURTH ROOT
U+02308	⌈	\lceil	LEFT CEILING
U+0230A	⌋	\lfloor	LEFT FLOOR
U+0231C	⌵	\ulcorner	UPPER LEFT CORNER
U+0231E	⌷	\llcorner	LOWER LEFT CORNER LIGHT LEFT TORTOISE SHELL BRACKET
U+02772		\lbrbrak	ORNAMENT
U+027C5	⌢	\lbag	LEFT S-SHAPED BAG DELIMITER
U+027CC	)	\longdivision	LONG DIVISION MATHEMATICAL LEFT WHITE SQUARE
U+027E6	⌌	\lBrack	BRACKET
U+027E8	<	\langle	MATHEMATICAL LEFT ANGLE BRACKET MATHEMATICAL LEFT DOUBLE ANGLE
U+027EA	⌠	\lAngle	BRACKET MATHEMATICAL LEFT WHITE TORTOISE
U+027EC		\lbrbrak	SHELL BRACKET
U+02983	⌈	\lBrace	LEFT WHITE CURLY BRACKET
U+02985	(	\lParen	LEFT WHITE PARENTHESIS
U+02987	⌈	\llparenthesis	Z NOTATION LEFT IMAGE BRACKET
U+02989	⌈	\llangle	Z NOTATION LEFT BINDING BRACKET
U+0298B	⌈	\lbrackubar	LEFT SQUARE BRACKET WITH UNDERBAR LEFT SQUARE BRACKET WITH TICK IN TOP
U+0298D	⌈	\lbrackultick	CORNER LEFT SQUARE BRACKET WITH TICK IN
U+0298F	⌈	\lbracklltick	BOTTOM CORNER
U+02991	⌈	\langedot	LEFT ANGLE BRACKET WITH DOT
U+02993	⌈	\lparenless	LEFT ARC LESS-THAN BRACKET
U+02997	⌈	\lblkbrbrak	LEFT BLACK TORTOISE SHELL BRACKET
U+029D8	⌈	\lvzigzag	LEFT WIGGLY FENCE
U+029DA	⌈	\lvzigzag	LEFT DOUBLE WIGGLY FENCE
U+029FC	⌈	\lcurvyangle	LEFT POINTING CURVED ANGLE BRACKET
U+03014		\lbrbrak	LEFT BROKEN BRACKET



U+03018                      \Lbrbrak                      LEFT WHITE TORTOISE SHELL BRACKET

And \mathclose:





USV	Ex.	Macro	Description
U+00029	)	\rparen	RIGHT PARENTHESIS
U+0005D	]	\rbrack	RIGHT SQUARE BRACKET
U+0007D	}	\rbrace	RIGHT CURLY BRACKET
U+000BB	»	\guillemotright	DOUBLE ANGLE QUOTATION MARK (GUILLEMET), RIGHT
U+02019	'	\rq	SINGLE QUOTATION MARK, RIGHT
U+0201B	‘	\quotsinglright	RISING SINGLE QUOTE, RIGHT (HIGH)
U+0201F	“	\quotdblright	RISING DOUBLE QUOTE, RIGHT (HIGH) SINGLE ANGLE QUOTATION MARK
U+0203A	>	\guilsinglright	(GUILLEMET), RIGHT
U+02309	]̂	\rceil	RIGHT CEILING
U+0230B	]̇	\rfloor	RIGHT FLOOR
U+0231D	]̋	\urcorner	UPPER RIGHT CORNER
U+0231F	]̌	\lrcorner	LOWER RIGHT CORNER
U+02773		\rbrbrak	ORNAMENT
U+027C6	⌋	\rbag	RIGHT S-SHAPED BAG DELIMITER MATHEMATICAL RIGHT WHITE SQUARE
U+027E7	⌋̂	\rBrack	BRACKET
U+027E9	⌋̇	\rangle	MATHEMATICAL RIGHT ANGLE BRACKET MATHEMATICAL RIGHT DOUBLE ANGLE
U+027EB	⌋̋	\rAngle	BRACKET MATHEMATICAL RIGHT WHITE TORTOISE
U+027ED		\Rbrbrak	SHELL BRACKET
U+02984	⌋̌	\rBrace	RIGHT WHITE CURLY BRACKET
U+02986	⌋̍	\rParen	RIGHT WHITE PARENTHESIS
U+02988	⌋̎	\rrparenthesis	Z NOTATION RIGHT IMAGE BRACKET
U+0298A	⌋̏	\rrangle	Z NOTATION RIGHT BINDING BRACKET RIGHT SQUARE BRACKET WITH
U+0298C	⌋̐	\rbrackubar	UNDERBAR RIGHT SQUARE BRACKET WITH TICK IN
U+0298E	⌋̑	\rbracklrtick	BOTTOM CORNER RIGHT SQUARE BRACKET WITH TICK IN
U+02990	⌋̒	\rbrackurtick	TOP CORNER
U+02992	⌋̓	\rangledot	RIGHT ANGLE BRACKET WITH DOT
U+02994	⌋̔	\rpangtr	RIGHT ARC GREATER-THAN BRACKET
U+02998	⌋̕	\rblbrbrak	RIGHT BLACK TORTOISE SHELL BRACKET
U+029D9	⌋̖	\rvzigzag	RIGHT WIGGLY FENCE
U+029DB	⌋̗	\Rvzigzag	RIGHT DOUBLE WIGGLY FENCE RIGHT POINTING CURVED ANGLE
U+029FD	⌋̘	\rcurvyangle	BRACKET

U+03015	<code>\rbrbrak</code>	RIGHT BROKEN BRACKET
U+03019	<code>\Rbrbrak</code>	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	́	<code>\acute</code>	ACUTE ACCENT
U+00302	̂	<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	̊	<code>\ocirc</code>	RING
U+0030C	̌	<code>\check</code>	CARON
U+00310	̎	<code>\candra</code>	CANDRABINDU (NON-SPACING)
U+00312	̐	<code>\oturnedcomma</code>	COMBINING TURNED COMMA ABOVE GREEK PSILI (SMOOTH BREATHING)
U+00313	̑	<code>\osmooth</code>	(NON-SPACING) GREEK DASIA (ROUGH BREATHING)
U+00314	̒	<code>\orough</code>	(NON-SPACING)
U+00315	̓	<code>\ocommatopright</code>	COMBINING COMMA ABOVE RIGHT
U+0031A	̔	<code>\droang</code>	LEFT ANGLE ABOVE (NON-SPACING)
U+020D0	̕	<code>\leftharpoonaccent</code>	COMBINING LEFT HARPOON ABOVE
U+020D1	̖	<code>\rightharpoonaccent</code>	COMBINING RIGHT HARPOON ABOVE COMBINING LONG VERTICAL LINE
U+020D2	̗	<code>\vertoverlay</code>	OVERLAY
U+020D6	̘	<code>\overleftarrow</code>	COMBINING LEFT ARROW ABOVE
U+020D7	̙	<code>\vec</code>	COMBINING RIGHT ARROW ABOVE
U+020DB	̚	<code>\dddots</code>	COMBINING THREE DOTS ABOVE
U+020DC	̛	<code>\ddddots</code>	COMBINING FOUR DOTS ABOVE
U+020E1	̜	<code>\overleftrightharpoon</code>	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7	̟	<code>\annuity</code>	COMBINING ANNUITY SYMBOL
U+020E8	̠	<code>\threeunderdot</code>	COMBINING TRIPLE UNDERDOT
U+020E9	̡	<code>\widebridgeabove</code>	COMBINING WIDE BRIDGE ABOVE COMBINING RIGHTWARDS HARPOON
U+020EC	̣	<code>\underrightharpoondown</code>	WITH BARB DOWNWARDS

U+020ED		<code>\underleftharpoondown</code>	COMBINING LEFTWARDS HARPOON WITH BARB DOWNWARDS
U+020EE		<code>\underleftarrow</code>	COMBINING LEFT ARROW BELOW
U+020EF		<code>\underrightarrow</code>	COMBINING RIGHT ARROW BELOW
U+020F0		<code>\asteraccent</code>	COMBINING ASTERISK ABOVE

---

## 7 Font features

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

```

570 \newcommand\um@zf@feature[2]{
571   \define@key[zf]{options}{#1}[] {
572     \ifum@fontspec@feature
573       #2
574     \else
575       \PackageError{fontspec/unicode-math}
576         {The ‘#1’ font feature can only be used for maths fonts}
577         {The feature you tried to use can only be in commands
578          like \protect\setmathfont}
579     \fi
580   }
581 }
```

### 7.1 OpenType maths font features

```

582 \um@zf@feature{ScriptStyle}{
583   \zf@update@ff{+ssty=0}
584 }
585 \um@zf@feature{ScriptScriptStyle}{
586   \zf@update@ff{+ssty=1}
587 }
```

### 7.2 Script and scriptscript font options

```

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

```

### 7.3 Range processing

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

```

592 \define@choicekey+[um]{options}{Range}[\@tempa\@tempb]{ALL}{
593   \ifcase\@tempb\relax
594     \global\let\um@char@range\@empty

```

```

595 \fi
596 }{
597 \xdef\um@char@range{#1}
598 }

```

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.

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

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

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

```

599 \newcommand\um@parse@term[4]{
600 \clist_map_variable:NNn \um@char@range \@ii {
601 \unless\ifx\@ii@empty
602 \@tempswafalse

```

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

```

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

```

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

```

612     \else
613         \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
614     \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.

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

```

‘r’ or ‘a’ or ‘b’ is included ‘r’ or ‘b’ or ‘c’ is  
included ‘3’ or ‘a’ or ‘b’ is included ‘3’ or ‘a’  
or ‘b’ is included

```

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

```

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

```

629 \def\um@parse@range#1-#2-#3\@nil#4\@nil{
630     \def\@tempa{#1}
631     \def\@tempb{#2}

```

Range	<b>r = x</b>
C-list input	\@ii=X
Macro input	\um@parse@range X-\@marker-\@nil#1\@nil
Arguments	<b>#1-#2-#3 = X-\@marker-{} </b>

```

632     \expandafter\ifx\expandafter\@marker\@tempb\relax
633         \ifnum#4=#1\relax
634             \@tempwatrue
635         \fi
636     \else

```

---

Range  $r \geq x$   
 C-list input  $\backslash @ii=X-$   
 Macro input  $\backslash um@parse@range X--\backslash @marker-\backslash @nil\#1\backslash @nil$   
 Arguments  $\#1-\#2-\#3 = X-\{\}-\backslash @marker-$

---

```

637 \ifx\@empty\@tempb
638 \ifnum#4>\numexpr#1-1\relax
639 \@tempswatrue
640 \fi
641 \else

```

---

Range  $r \leq y$   
 C-list input  $\backslash @ii=-Y$   
 Macro input  $\backslash um@parse@range -Y-\backslash @marker-\backslash @nil\#1\backslash @nil$   
 Arguments  $\#1-\#2-\#3 = \{\}-Y-\backslash @marker-$

---

```

642 \ifx\@empty\@tempa
643 \ifnum#4<\numexpr#2+1\relax
644 \@tempswatrue
645 \fi

```

---

Range  $x \leq r \leq y$   
 C-list input  $\backslash @ii=X-Y$   
 Macro input  $\backslash um@parse@range X-Y-\backslash @marker-\backslash @nil\#1\backslash @nil$   
 Arguments  $\#1-\#2-\#3 = X-Y-\backslash @marker-$

---

```

646 \else
647 \ifnum#4>\numexpr#1-1\relax
648 \ifnum#4<\numexpr#2+1\relax
649 \@tempswatrue
650 \fi
651 \fi
652 \fi
653 \fi
654 \fi
655 }

```

$\backslash um@setmathcode$  #1 : Starting input char(s)

#2 : Number of iterations

#3 : Starting output char

Loops through character ranges setting  $\backslash mathcode$ .

```

656 \newcommand\um@setmathcode[3][1]{
657 \clist_map_variable:nNn {#2} \l_um_input_num {
658 \prg_stepwise_variable:nnnNn{1}{1}{#1} \l_um_incr_num {
659 \SetMathCode
660 {\numexpr \l_um_incr_num+ \l_um_input_num - 1\relax}
661 {\mathalpha}{\um@symfont}
662 {\numexpr \l_um_incr_num + #3 - 1\relax}
663 }
664 }

```

665 }

\um\_set\_mathalphabet\_char:Nnnn #1 : Maths alphabet

#2 : Input char(s)

#3 : Output char

Loops through character ranges setting \mathcode.

666 \cs\_set:Npn \exp\_args:Nnff {\::n\::f\::f\:::}

667 \cs\_new:Nn \um\_set\_mathalphabet\_char:Nnn {

668 \clist\_map\_variable:nNn {#2} \l\_um\_input\_num {

669 \exp\_args:Nnff \um\_mathmap:Nnn {#1}

670 {\number\numexpr\l\_um\_input\_num\relax} {\number\numexpr#3\relax}

671 }

672 }

\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.

673 \cs\_new:Nn \um\_set\_mathalph\_range:nNnn {

674 \clist\_map\_variable:nNn {#3} \l\_um\_input\_num {

675 \prg\_stepwise\_variable:nnnNn {0}{1}{#1} \l\_um\_inc\_num {

676 \exp\_args:Nnff \um\_mathmap:Nnn {#2}

677 {\number\numexpr \l\_um\_inc\_num + \l\_um\_input\_num \relax}

678 {\number\numexpr \l\_um\_inc\_num + #4 \relax}

679 }

680 }

681 }

682 \cs\_new:Nn \um\_set\_mathalphabet\_numbers:Nnn {

683 \um\_set\_mathalph\_range:nNnn {9}{#1}{#2}{#3}

684 }

685 \cs\_new:Nn \um\_set\_mathalphabet\_latin:Nnn {

686 \um\_set\_mathalph\_range:nNnn {25}{#1}{#2}{#3}

687 }

688 \cs\_new:Nn \um\_set\_mathalphabet\_greek:Nnn {

689 \um\_set\_mathalph\_range:nNnn {24}{#1}{#2}{#3}

690 }

---

BCDBCD ABCDEF

`{\um@setmathcode[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.

```

691 \AtBeginDocument{\um@resolve@greek}
692 \newcommand\um@resolve@greek{
693   \def\Alpha{\mitAlpha}
694   \def\Beta{\mitBeta}
695   \def\Gamma{\mitGamma}
696   \def\Delta{\mitDelta}
697   \def\Epsilon{\mitEpsilon}
698   \def\Zeta{\mitZeta}
699   \def\Eta{\mitEta}
700   \def\Theta{\mitTheta}
701   \def\Iota{\mitIota}
702   \def\Kappa{\mitKappa}
703   \def\Lambda{\mitLambda}
704   \def\Mu{\mitMu}
705   \def\Nu{\mitNu}
706   \def\Xi{\mitXi}
707   \def\Omicron{\mitOmicron}
708   \def\Pi{\mitPi}
709   \def\Rho{\mitRho}
710   \def\varTheta{\mitvarTheta}
711   \def\Sigma{\mitSigma}
712   \def\Tau{\mitTau}
713   \def\Upsilon{\mitUpsilon}
714   \def\Phi{\mitPhi}
715   \def\Chi{\mitChi}
716   \def\Psi{\mitPsi}
717   \def\Omega{\mitOmega}

```

Lowercase:

```

718   \def\alpha{\mitalpha}
719   \def\beta{\mitbeta}
720   \def\gamma{\mitgamma}
721   \def\delta{\mitdelta}
722   \def\epsilon{\mitepsilon}
723   \def\zeta{\mitzeta}
724   \def\eta{\miteta}
725   \def\theta{\mittheta}
726   \def\iota{\mitiota}
727   \def\kappa{\mitkappa}
728   \def\lambda{\mitlambda}
729   \def\mu{\mitmu}
730   \def\nu{\mitnu}
731   \def\xi{\mitxi}
732   \def\omicron{\mitomicron}
733   \def\pi{\mitpi}
734   \def\rho{\mitrho}
735   \def\varsigma{\mitvarsigma}

```



```

736 \def\sigma{\mitsigma}
737 \def\tau{\mittau}
738 \def\upsilon{\mitupsilon}
739 \def\phi{\mitphi}
740 \def\chi{\mitchi}
741 \def\psi{\mitpsi}
742 \def\omega{\mitomega}
743 \def\varepsilon{\mitvarepsilon}
744 \def\vartheta{\mitvartheta}
745 \def\varkappa{\mitvarkappa}
746 \def\varphi{\mitvarphi}
747 \def\varrho{\mitvarrho}
748 \def\varpi{\mitvarpi}
749 }

```

\um@def@numbers

```

750 \newcommand\um@def@numbers{
751   \um@setmathcode[10]{\um@usv@num}{\um@usv@num}
752 }

```

\um\_setup\_literals: : TODO : other literal symbols

```

753 \cs_set:Nn \um_setup_literals: {
754   \um@setmathcode[26]{\um@usv@upLatin}{\um@usv@upLatin}
755   \um@setmathcode[26]{\um@usv@itLatin}{\um@usv@itLatin}
756   \um@setmathcode[26]{\um@usv@itlatin}{\um@usv@itlatin}
757   \um@setmathcode{\um@usv@ith}{\um@usv@ith}
758   \um@setmathcode[26]{\um@usv@uplatin}{\um@usv@uplatin}
759   \um@setmathcode[25]{\um@usv@upGreek}{\um@usv@upGreek}
760   \um@setmathcode{\um@usv@varTheta}{\um@usv@varTheta}
761   \um@setmathcode[25]{\um@usv@itGreek}{\um@usv@itGreek}
762   \um@setmathcode[25]{\um@usv@upgreek}{\um@usv@upgreek}
763 }

```

\um\_setup\_bf\_literals: TODO: other literal symbols

```

764 \cs_set:Nn \um_setup_bf_literals: {
765   \um@setmathcode[26]{\um@usv@bfLatin}{\um@usv@bfLatin}
766   \um@setmathcode[26]{\um@usv@bflatin}{\um@usv@bflatin}
767   \um@setmathcode[26]{\um@usv@bfitLatin}{\um@usv@bfitLatin}
768   \um@setmathcode[26]{\um@usv@bfitlatin}{\um@usv@bfitlatin}
769   \um@setmathcode[25]{\um@usv@bfGreek}{\um@usv@bfGreek}
770   \um@setmathcode[25]{\um@usv@bfgreek}{\um@usv@bfgreek}
771   \um@setmathcode[25]{\um@usv@bfitGreek}{\um@usv@bfitGreek}
772   \um@setmathcode[25]{\um@usv@bfitgreek}{\um@usv@bfitgreek}
773 }

```

\um@def@upLatin

```

774 \newcommand\um@def@upLatin{
775   \um@setmathcode[26]{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
776 }

\um@def@itLatin
777 \newcommand\um@def@itLatin{
778   \um@setmathcode[26]{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
779 }

\um@def@itlatin Don't overlook 'h', which maps to U+210E: PLANCK CONSTANT instead of the ex-
pected U+1D455: MATHEMATICAL ITALIC SMALL H.
780 \newcommand\um@def@itlatin{
781   \um@setmathcode[26]{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
782   \um@setmathcode{\`h,\um@usv@ith}{\um@usv@ith}
783 }

\um@def@uplatin
784 \newcommand\um@def@uplatin{
785   \um@setmathcode[26]{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
786   \um@setmathcode{\um@usv@ith}{\`h}
787 }

\um@def@upGreek
788 \newcommand\um@def@upGreek{
789   \um@setmathcode[25]{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
790   \um@setmathcode{\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
791 }

\um@def@itGreek
792 \newcommand\um@def@itGreek{
793   \um@setmathcode[25]{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
794   \um@setmathcode{\um@usv@varTheta}{\um@usv@itvarTheta}
795 }

\um@def@upgreek
796 \newcommand\um@def@upgreek{
797   \um@setmathcode[25]{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
798   \um@setmathcode{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
799   \um@setmathcode{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
800   \um@setmathcode{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
801   \um@setmathcode{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
802   \um@setmathcode{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
803   \um@setmathcode{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
804 }

```

`\um@def@itgreek`

```
805 \newcommand\um@def@itgreek{
806   \um@setmathcode[25]{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
807   \um@setmathcode{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
808   \um@setmathcode{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
809   \um@setmathcode{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
810   \um@setmathcode{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
811   \um@setmathcode{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
812   \um@setmathcode{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
813 }
```

## 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.

```
814 \cs_new:Nn \um_setup_math_alphabet:n {
815   \um_glyph_if_exist:nTF {\csname um@usv@#1latin \endcsname}{
816     \um_maybe_init_alphabet:n {#1}
817     \um_prepare_alph:n {#1}
818     \use:c {um_config_math#1:}
819   }{
820     \um@PackageWarning{Math~ alphabet~ "#1"~ not~ found~ with~ this~ font}
821   }
822 }

823 \cs_set:Nn \um_init_alphabet:n {
824   \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
825 }
```

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

```
826 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
827   \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
828 }
```

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

```

829 \cs_new:Nn \um_prepare_alph:n {
830   \cs_if_exist:cF {um_math#1:n} {
831     \cs_set:cpn {um_math#1:n} ##1 {
832       \begingroup \use:c {um_setup_math#1:} ##1 \endgroup
833     }
834     \cs_set_protected:cpn {math#1} {
835       \mode_if_math:F {
836         \expandafter\non@alpherr\expandafter{\csname math#1\endcsname\space}
837       }
838       \use:c {um_math#1:n}
839     }
840   }
841 }

842 \cs_new:Nn \um_setup_alphabets: {
843   \um_setup_math_alphabet:n {up}   }
844   \um_setup_math_alphabet:n {it}   }
845   \um_setup_math_alphabet:n {bb}   }
846   \um_setup_math_alphabet:n {scr}  }
847   \um_setup_math_alphabet:n {frac} }
848   \um_setup_math_alphabet:n {sf}   }
849   \um_setup_math_alphabet:n {sfit} }
850   \um_setup_math_alphabet:n {tt}   }
851   \um_setup_math_alphabet:n {bf}   }
852   \um_setup_math_alphabet:n {bfup} }
853   \um_setup_math_alphabet:n {bfit} }
854   \um_setup_math_alphabet:n {bfscr} }
855   \um_setup_math_alphabet:n {bffrac} }
856   \um_setup_math_alphabet:n {bfsf} }
857   \um_setup_math_alphabet:n {bfsfup} }
858   \um_setup_math_alphabet:n {bfsfit} }
859 }

```

: TODO : nested alphabets?

### 8.0.1 Upright: `\mathup`

[illegible]

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

```

860 \cs_new:Npn \um_config_mathup: {
861   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
862   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
863   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
864   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
865   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}
866   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}
867   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varTheta}
868   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
869   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
870   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
871   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
872   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
873   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
874 }

```

## 8.0.2 Italic: \mathit

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

### Roman:

```

875 \cs_new:Npn \um_config_mathit: {
876   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
877   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
878   \um_set_mathalphabet_char:Nnn{\mathit}{\h,\um@usv@ith}{\um@usv@ith}

```

### Greek:

```

879   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
880   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
881   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
882   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartial}
883   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarThe}
884   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itva}
885   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvarthe}
886   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkap}
887   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
888   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
889   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
890 }

```

### 8.0.3 Blackboard or double-struck: `\mathbb`

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

Numbers:

```
891 \cs_new:Npn \um_config_mathbb: {
892   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@num}{\um@usv@bbnum}
```

Roman uppercase:

```
893 \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
894 \um_set_mathalphabet_char:Nnn{\mathbb}{`\C,"1D60A}{`"2102}
895 \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D60F}{`"210D}
896 \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D60F}{`"2115}
897 \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D617}{`"2119}
898 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D618}{`"211A}
899 \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D619}{`"211D}
900 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D621}{`"2124}
```

Roman lowercase:

```
901 \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}
902 }
```

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

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

```
903 \cs_new:Npn \um_config_mathscr: {
904   \um_set_mathalphabet_latin:Nnn{\mathscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}
905   \um_set_mathalphabet_char:Nnn{\mathscr}{`\B,"1D435}{`"212C}
906   \um_set_mathalphabet_char:Nnn{\mathscr}{`\E,"1D438}{`"2130}
907   \um_set_mathalphabet_char:Nnn{\mathscr}{`\F,"1D439}{`"2131}
908   \um_set_mathalphabet_char:Nnn{\mathscr}{`\H,"1D43B}{`"210B}
909   \um_set_mathalphabet_char:Nnn{\mathscr}{`\I,"1D43C}{`"2110}
910   \um_set_mathalphabet_char:Nnn{\mathscr}{`\L,"1D43F}{`"2112}
911   \um_set_mathalphabet_char:Nnn{\mathscr}{`\M,"1D440}{`"2133}
912   \um_set_mathalphabet_char:Nnn{\mathscr}{`\R,"1D445}{`"211B}
913   \um_set_mathalphabet_latin:Nnn{\mathscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrlatin}
914   \um_set_mathalphabet_char:Nnn{\mathscr}{`\e,"1D452}{`"212F}
915   \um_set_mathalphabet_char:Nnn{\mathscr}{`\g,"1D454}{`"210A}
916   \um_set_mathalphabet_char:Nnn{\mathscr}{`\o,"1D45C}{`"2134}
917 }
```

## 8.0.5 Fraktur or fraktur or blackletter: `\mathfrak`

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

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

Letters, with exceptions  $\{\mathfrak{C}, \mathfrak{H}, \mathfrak{I}, \mathfrak{R}, \mathfrak{Z}\}$ :

```

918 \cs_new:Npn \um_config_mathfrak: {
919   \um_set_mathalphabet_latin:Nnn{\mathfrak}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLatin}
920   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\C,"1D436}{`"212D}
921   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\H,"1D43B}{`"210C}
922   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\I,"1D43C}{`"2111}
923   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\R,"1D445}{`"211C}
924   \um_set_mathalphabet_char:Nnn{\mathfrak}{`\Z,"1D44D}{`"2128}
925   \um_set_mathalphabet_latin:Nnn{\mathfrak}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklatin}
926 }

```

## 8.0.6 Sans serif: `\mathsf`

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

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

```

927 \cs_new:Npn \um_config_mathsf: {
928   \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnum}
929   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfLatin}
930   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sflatin}
931 }

```

## 8.0.7 Sans serif italic: `\mathsf`

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

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

```

932 \cs_new:Npn \um_config_mathsf: {
933   \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnum}
934   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfLatin}
935   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sflatin}
936 }

```

## 8.0.8 Typewriter or monospaced: `\mathtt`

0123456789	<code>\$\mathtt{0123456789}\$ \\\</code>
ABCDEFGHIJKLMN O PQRSTUVWXYZ	<code>\$\mathtt{ABCDEFGHIJKLMN O PQRSTUVWXYZ}\$ \\\</code>
abcdefghijklmn opqrstuvwxyz	<code>\$\mathtt{abcdefghijklmn opqrstuvwxyz}\$ \\\</code>

```

937 \cs_new:Npn \um_config_mathtt: {
938   \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
939   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
940   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlatin}
941 }

```

## 8.1 Bold alphabets' character mappings

### 8.1.1 Bold: `\mathbf`

0123456789	<code>\$\mathbf{0123456789}\$ \\\</code>
ABCDEFGHIJKLMN O PQRSTUVWXYZ	<code>\$\mathbf{ABCDEFGHIJKLMN O PQRSTUVWXYZ}\$ \\\</code>
abcdefghijklmn opqrstuvwxyz	<code>\$\mathbf{abcdefghijklmn opqrstuvwxyz}\$ \\\</code>
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ	<code>\$\mathbf{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}\$ \\\</code>
Θ	<code>\$\mathbf{Θ}\$ \\\</code>
αβγδεζηθικλμνξοπρστυφχψω	<code>\$\mathbf{αβγδεζηθικλμνξοπρστυφχψω}\$ \\\</code>
εθκφρϖ	<code>\$\mathbf{εθκφρϖ}\$ \\\</code>

```

942 \cs_new:Npn \um_config_mathbf: {
943   \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
944   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
945   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
946   \if@um@bfliteral
947     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfLatin}
948     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
949     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@bflatin}
950     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@bfitlatin}
951     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfGreek}
952     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
953     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfgreek}
954     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitegreek}
955     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
956     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
957     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
958     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
959     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
960     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bfvarepsilon}

```



```

961 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
962 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
963 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
964 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
965 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}
966 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
967 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
968 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
969 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
970 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
971 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
972 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
973 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
974 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
975 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
976 \else
977 \ifum@bfupLatin
978 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bLatin}
979 \else
980 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
981 \fi
982 \ifum@bfuplatin
983 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@blatin}
984 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
985 \else
986 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
987 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
988 \fi
989 \ifum@bfupGreek
990 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfGreek}
991 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarTheta}
992 \else
993 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
994 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
995 \fi
996 \ifum@bfupgreek
997 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfgreek}
998 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
999 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1000 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1001 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1002 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1003 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1004 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartial}
1005 \else
1006 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}

```

```

1007 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@b
1008 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitv
1009 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitv
1010 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarph
1011 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho
1012 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1013 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpar
1014 \fi
1015 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla_up_or_it
1016 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpartial_up_
1017 \fi
1018 }

```

### 8.1.2 Bold Italic: \mathbfit

<b>0123456789</b>	<code>\$\mathbfit{0123456789}\$ \\\</code>
<b>ABCDEFGHIJKLMNOPQRSTUVWXYZ</b>	<code>\$\mathbfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\\</code>
<b>abcdefghijklmnopqrstuvwxyz</b>	<code>\$\mathbfit{abcdefghijklmnopqrstuvwxyz}\$ \\\</code>
<b>ΑΒΓΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ</b>	<code>\$\mathbfit{ΑΒΓΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ}\$\quad</code>
<b>αβγδεζηθικλμνξοπρστυφχψω εθκφρψ</b>	<code>\$\mathbfit{αβγδεζηθικλμνξοπρστυφχψω}\$\quad</code>
	<code>\$\mathbfit{0000000000000000π000000000}\$\quad</code>
	<code>\$\mathbfit{0000000}\$ \\\</code>

```

1019 \cs_new:Npn \um_config_mathbfit: {
1020   \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnun}
1021   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1022   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
1023   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
1024   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}
1025   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfLatin}{\um@usv@bfitLatin}
1026   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bflatin}{\um@usv@bfitlatin}
1027   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfGreek}{\um@usv@bfitGreek}
1028   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfgreek}{\um@usv@bfitgreek}
1029   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitva
1030   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}
1031   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpart
1032   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf
1033   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitva
1034   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitva
1035   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi
1036   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho
1037   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1038 }

```

### 8.1.3 Bold Italic: `\mathbfup`

0123456789  
ABCDEFGHIJKLMNOPQRSTUVWXYZ  
abcdefghijklmnopqrstuvwxyz  
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ  
αβγδεζηθικλμνξοπρστυφχψω εθκφρσ

[illegible]

```

1039 \cs_new:Npn \um_config_mathbfup: {
1040   \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bnum}
1041   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bLatin}
1042   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@blatin}
1043   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bGreek}
1044   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfgreek}
1045   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bLatin}{\um@usv@bLatin}
1046   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@blatin}{\um@usv@blatin}
1047   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bGreek}{\um@usv@bGreek}
1048   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfgreek}{\um@usv@bfgreek}
1049   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarTheta}
1050   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}
1051   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartial}
1052   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
1053   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1054   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1055   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1056   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1057   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1058 }

```

#### 8.1.4 Bold fractur or fraktur or blackletter: `\mathbf{frak}`

၁၂၃၄၅၆၇၈၉၁၀၁၁၂၁၃၁၄၁၅၁၆၁၇၁၈၁၉၂၀  
 abcdefghijklmnopqrstuvwxyz

```
\setmathfont{Cambria Math}

$$\frac{ABCDEFGHIJKLMNOPQRSTUVWXYZ}{abcdefghijklmnopqrstuvwxyz}$$

```

```

1059 \cs_new:Npn \um_config_mathbbfrak: {
1060   \um_set_mathalphabet_numbers:Nnn{\mathbbfrak}{\um@usv@num}{\um@usv@bnum}
1061   \um_set_mathalphabet_latin:Nnn{\mathbbfrak}{\um@usv@upLatin, \um@usv@itLatin, \um@usv@frakLat
1062   \um_set_mathalphabet_latin:Nnn{\mathbbfrak}{\um@usv@uplatin, \um@usv@itlatin, \um@usv@fraklati
1063 }

```

### 8.1.5 Bold script or calligraphic: `\mathbf{scr}`

*ABCDEFGHIJKLMNOPQRSTUVWXYZ  
abcdefghijklmnopqrstuvwxyz*

```
\setmathfont{Cambria Math}
$\mathbf{fscr}{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \
$\mathbf{fscr}{abcdefghijklmnopqrstuvwxyz}$ \
```

```

1064 \cs_new:Npn \um_config_mathbfscr: {
1065   \um_set_mathalphabet_numbers:Nnn{\mathbfscr}{\um@usv@num}{\um@usv@bfnum}
1066   \um_set_mathalphabet_latn:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrLat
1067   \um_set_mathalphabet_latn:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrLat
1068 }

```

### 8.1.6 Bold sans serif: `\mathbfsf`

0123456789  
ABCDEFGHIJKLMNOPQRSTUVWXYZ  
abcdefghijklmnopqrstuvwxyz  
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ  
αβγδεζηθικλμνξοπρστυφχψω ειδικρω

[illegible]

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

```

1069 \cs_new:Npn \um_config_mathbfsf: {
1070   \um_set_mathalphabet_numbers:Nnn{\mathbfsf}{\um@usv@num}{\um@usv@bfnun}
1071   \um_set_mathalphabet_latin:Nnn{\mathbfsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfLatin}
1072   \um_set_mathalphabet_latin:Nnn{\mathbfsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsflatin}
1073   \um_set_mathalphabet_greek:Nnn{\mathbfsf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfGreek}
1074   \um_set_mathalphabet_greek:Nnn{\mathbfsf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfgreek}

```

Others:

```

1075 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@varTheta,\um@usv@itvarTheta}{ "1D767}
1076 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@Nabla,\um@usv@itNabla}{ "1D76F}
1077 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@partial,\um@usv@itpartial}{ "1D789}
1078 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@varepsilon,\um@usv@itvarepsilon}{ "1D78A}
1079 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@vartheta,\um@usv@itvartheta}{ "1D78B}
1080 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@varkappa,\um@usv@itvarkappa}{ "1D78C}
1081 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@varphi,\um@usv@itvarphi}{ "1D78D}
1082 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@varrho,\um@usv@itvarrho}{ "1D78E}
1083 \um_set_mathalphabet_char:Nnn{\mathbfsf}\um@usv@varpi,\um@usv@itvarpi}{ "1D78F}
1084 }

```

0123456789	$\setmathfont{\STIXGeneral-Bold}$
DEFGHIJKLMNOPQRSTUVWXYZ	$\mathbf{\text{\texttt{\textbackslash mathbbsfup\{0123456789\}}}}$
abcdefghijklmnopqrstuvwxyz	$\mathbf{\text{\texttt{\textbackslash mathbbsfup\{ABCDEFGHIJKLMNOPQRSTUVWXYZ\}}}}$
ΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	$\mathbf{\text{\texttt{\textbackslash mathbbsfup\{abcdefghijklmnopqrstuvwxyz\}}}}$
δεζηθικλμνξοπρστυφχψω εδκφρω	$\mathbf{\text{\texttt{\textbackslash mathbbsfup\{ΑΒΓΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ\}}}\quad\mathbf{\text{\texttt{\textbackslash mathbbsfup\{\}\}}}$
	$\mathbf{\text{\texttt{\textbackslash mathbbsfup\{\pi\}}}}$
	$\mathbf{\text{\texttt{\textbackslash mathbbsfup\{\pi\}}}}$

```

1085 \cs_new:Npn \um_config_mathbbsfup: {
1086   \um_set_mathalphabet_numbers:Nnn{\mathbbsfup}{\um@usv@num}{\um@usv@bfnun}
1087   \um_set_mathalphabet_latin:Nnn{\mathbbsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfLat
1088   \um_set_mathalphabet_latin:Nnn{\mathbbsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsflat
1089   \um_set_mathalphabet_greek:Nnn{\mathbbsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfGre
1090   \um_set_mathalphabet_greek:Nnn{\mathbbsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfgre

```

```

1091 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varTheta,\um@usv@itvarTheta}{~"1D767}
1092 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@Nabla,\um@usv@itNabla}{~"1D76F}
1093 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@partial,\um@usv@itpartial}{~"1D789}
1094 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{~"1D78A}
1095 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@vartheta,\um@usv@itvartheta}{~"1D78B}
1096 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varkappa,\um@usv@itvarkappa}{~"1D78C}
1097 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varphi,\um@usv@itvarphi}{~"1D78D}
1098 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varrho,\um@usv@itvarrho}{~"1D78E}
1099 \um_set_mathalphabet_char:Nnn{\mathbfsup}{\um@usv@varpi,\um@usv@itvarpi}{~"1D78F}
1100 }

```

<i>ABCDEFGHIJKLMN</i>	$\backslash\mathrm{setmathfont}\{\mathrm{STIXGeneral}\text{-}\mathrm{BoldItalic}\}$
<i>OPQRSTUVWXYZ</i>	$\$\mathrm{mathbfsfit}\{\mathrm{0123456789}\}\$\ \backslash\backslash$
<i>abcdefghijklmnopqrstuvwxyz</i>	$\$\mathrm{mathbfsfit}\{\mathrm{ABCDEFGHIJKLMN}\}\$\ \backslash\backslash$
<i>ABCDEFGHIJKLMNOPQRSTUVWXYZ</i>	$\$\mathrm{mathbfsfit}\{\mathrm{abcdefghijklmnopqrstuvwxyz}\}\$\ \backslash\backslash$
<i>αβγδεζηθικλμνξοπρστυφχψω</i>	$\$\mathrm{mathbfsfit}\{\mathrm{ABΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}\}\$\ \backslash\backslash$
<i>αβγδεζηθικλμνξοπρστυφχψω</i>	$\$\mathrm{mathbfsfit}\{\mathrm{0}\}\$\ \backslash\backslash$
<i>εδϑϕρσ</i>	$\$\mathrm{mathbfsfit}\{\mathrm{000000000000000000000000}\}\$\ \backslash\backslash$
	$\$\mathrm{mathbfsfit}\{\mathrm{000000}\}\$\ \backslash\backslash$

53

Other symbols:

```

1107 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{1D7A1}
1108 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNabla}
1109 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsfitpartial}
1110 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7C4}
1111 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{1D7C5}
1112 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{1D7C6}
1113 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{1D7C7}
1114 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{1D7C8}
1115 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{1D7C9}
1116 }

```

## 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 `"ABCDEF` into the  $\text{\texttt{X}\texttt{Y}\texttt{Z}}$  ‘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.

```

1117 \begingroup
1118 \char_make_other:N \^
1119 \gdef\um@scancharlet#1="#2\@nil{
1120   \lowercase{
1121     \scantokens{\global\let#1=^^^^^#2}
1122   }
1123 }
1124 \gdef\um@scanactivedef"#1\@nil#2{
1125   \lowercase{
1126     \scantokens{\global\def^^^^^#1{#2}}
1127   }
1128 }
1129 \endgroup
1130 \let\unicodemathgobble\@gobble

```

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

```

1131 \begingroup
1132 \def\UnicodeMathSymbol#1#2#3#4{
1133   \um@scancharlet#2=#1\@nil
1134 }

```

```

1135 \input{unicode-math-table.tex}
1136 \endgroup

```

## 8.3 Epilogue

Lots of little things to tidy up.

### 8.3.1 Unicode radicals

Undo the damage made to `\sqrt`:

```

1137 \DeclareRobustCommand\sqrt{\ifnextchar[\@sqrt\sqrtsign}

```

### 8.3.2 Primes

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

```

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

```

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

```

U+2032: PRIME (\primesingle): x'
U+2033: DOUBLE PRIME (\primedouble): x''
U+2034: TRIPLE PRIME (\primetriple): 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 `sssty` feature is applied:

```

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

```

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

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

- Prime encountered; `pcount=1`.
- Scan ahead; if prime: `pcount:=pcount+1`; repeat.
- If not prime, stop scanning.

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

```

1138 \muskip_new:N \g_um_primekern_muskip
1139 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1140 \num_new:N \l_um_primecount_num

1141 \cs_new:Nn \um_nprimes:n {
1142   \primesingle
1143   \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1144 }
1145 \cs_new:Nn \um_nprimes_select:n {
1146   \prg_case_int:nnn {#1}{
1147     {1} { \primesingle }
1148     {2} {
1149       \um_glyph_if_exist:nTF {"2033} {\primedouble} {\um_nprimes:n {#1}}
1150     }
1151     {3} {
1152       \um_glyph_if_exist:nTF {"2034} {\primetriples} {\um_nprimes:n {#1}}
1153     }
1154     {4} {
1155       \um_glyph_if_exist:nTF {"2057} {\primequadruple} {\um_nprimes:n {#1}}
1156     }
1157   }{
1158     \um_nprimes:n {#1}
1159   }
1160 }

```

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?

```

1161 \cs_new:Nn \um_scanprime: {
1162   \bgroup
1163   \num_zero:N \l_um_primecount_num
1164   \um_scanprime_collect:
1165 }
1166 \cs_new:Nn \um_scanprime_collect: {
1167   \num_incr:N \l_um_primecount_num

```



```

1168 \peek_charcode_remove:NTF ' {
1169   \um_scanprime_collect:
1170 }{
1171   \peek_meaning_remove:NTF \um_scanprime: {
1172     \um_scanprime_collect:
1173   }{
1174     \peek_charcode_remove:NTF ^^^^2032 {
1175       \um_scanprime_collect:
1176     }{
1177       \um_nprimes_select:n {\l_um_primecount_num}
1178     }
1179   }
1180 }
1181 }
1182 }

1183 \cs_set_eq:NN \prime \um_scanprime:
1184 \group_begin:
1185   \char_make_active:N \prime
1186   \char_make_active:n {"2032}
1187   \cs_gset_eq:NN \prime \um_scanprime:
1188   \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1189 \group_end:

```

### 8.3.3 Radicals

`\r@@t` #1 : A mathstyle (for `\mathpalette`)  
 #2 : Leading superscript for the sqrt sign  
 A re-implementation of L<sup>A</sup>T<sub>E</sub>X's hard-coded n-root sign using the appropriate `\fontdimens`.

```

1190 \def\r@@t#1#2{
1191   \setbox\z@\hbox{$\math #1\sqrtsign{#2}$}
1192   \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1193   \raise \dimexpr(
1194     \um@fontdimen@percent{65}{\um@font}\ht\z@-
1195     \um@fontdimen@percent{65}{\um@font}\dp\z@
1196   )\relax
1197   \copy \rootbox
1198   \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1199   \box \z@
1200 }

```

### 8.3.4 Synonyms and all the rest

We need to change L<sup>A</sup>T<sub>E</sub>X's idea of the font used to typeset things like `\sin` and `\cos`:

```

1201 \def\operator@font{\um_setup_mathup:}
1202 \def\to{\rightarrow}
1203 \def\le{\leq}
1204 \def\ge{\geq}

\mathcal
1205 \def\mathcal{\mathscr}

\mathrm
1206 \def\mathrm{\mathup}

      Overriding amsmath definitions:
1207 \AtBeginDocument{
1208   \def\@cdots{\mathinner{\cdots}}
1209 }

      Interaction with beamer:
1210 \AtBeginDocument{
1211   \ifpackageloaded{beamer}{
1212     \ifbeamer@suppressreplacements\else
1213       \PackageWarningNoLine{unicode-math}{
1214         Disabling~ beamer's~ math~ setup.^{}
1215         Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1216       }
1217       \beamer@suppressreplacementstrue
1218     \fi
1219   }{}
1220 }

      The end.
1221 \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](http://ams.org/STIX)). A version is located at <http://www.ams.org/STIX/bnb/stix-tbl.asc> but check <http://www.ams.org/STIX/> for more up-to-date info.

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

A single file is produced containing all (more than 3298) symbols. Future optimisations might include generating various (possibly overlapping) subsets so

not all definitions must be read just to redefine a small range of symbols. Performance for now seems to be acceptable without such measures.

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

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

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

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

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

Print the actual entry corresponding to the unicode character:

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

Now replace the STIX class abbreviations with their  $\TeX$  macro names.

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

A 'fence' defined by the STIX table is something like `\vert`; in  $\text{X}\mathbb{T}\text{E}\text{X}$  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  $\{T1\}\{lmr\}\{m\}\{n\}$ .

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

```
\DeclareSymbolFont{\name}\langle NFSS decl.\rangle
```

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

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

```
\DeclareMathAlphabet{\cmd}\langle NFSS decl.\rangle
```

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\}\langle NFSS decl.\rangle
```

```
\SetMathAlphabet{\cmd}\{\maths version\}\langle NFSS decl.\rangle
```

**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  $\XTeX$ . 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 X<sub>Y</sub>TeX.

Accents are not included yet.

**Summary** For symbols, something like:

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

For characters, something like:

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

## File III

# X<sub>Y</sub>TeX math font dimensions

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

<code>\fontdimen</code>	Dimension name	Description
10	SCRIPTPERCENTSCALEDOWN	Percentage of scaling down for script level 1. Suggested value: 80%.
11	SCRIPTSCRIPTPERCENTSCALEDOWN	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	DELIMITEDSUBFORMULA-MINHEIGHT	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height $\times 1.5$ .
13	DISPLAYOPERATORMIN-HEIGHT	Minimum height of n-ary operators (such as integral and summation) for formulas in display mode.

\fontdimen	Dimension name	Description
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: $\frac{1}{5}$ x-height.
20	SUBSCRIPTBASELINEDROP-MIN	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	SUPERSCRIPSHIFTUP-CRAMPED	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.

\fontdimen	Dimension name	Description
24	SUPERSCRIPTBASELINEDROP-MAX	Maximum allowed drop of the baseline of superscripts relative to the (ink) top of the base. Checked for bases that are treated as a box or extended shape. Positive for superscript baseline below the base top.
25	SUBSUPERSCRIPGAPMIN	Minimum gap between the superscript and subscript ink. Suggested: 4×default rule thickness.
26	SUPERSCRIPBTOTTOMMAX-WITHSUBSCRIPT	The maximum level to which the (ink) bottom of superscript can be pushed to increase the gap between superscript and subscript, before subscript starts being moved down. Suggested: /5 x-height.
27	SPACEAFTERScript	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	UPPERLIMITBASELINERISE-MIN	Minimum distance between baseline of upper limit and (ink) top of the base operator.
30	LOWERLIMITGAPMIN	Minimum gap between (ink) top of the lower limit, and (ink) bottom of the base operator.
31	LOWERLIMITBASELINEDROP-MIN	Minimum distance between baseline of the lower limit and (ink) bottom of the base operator.
32	STACKTOPSHIFTUP	Standard shift up applied to the top element of a stack.
33	STACKTOPDISPLAYSTYLE-SHIFTUP	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	STACKBOTTOMDISPLAY-STYLESHIFTDOWN	Standard shift down applied to the bottom element of a stack in display style. Positive for moving in the downward direction.

\fontdimen	Dimension name	Description
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	STRETCHSTACKBOTTOM-SHIFTDOWN	Standard shift down applied to the bottom element of the stretch stack. Positive for moving in the downward direction.
40	STRETCHSTACKGAPABOVE-MIN	Minimum gap between the ink of the stretched element, and the (ink) bottom of the element above. Suggested: UpperLimitGapMin
41	STRETCHSTACKGAPBELOW-MIN	Minimum gap between the ink of the stretched element, and the (ink) top of the element below. Suggested: LowerLimitGapMin.
42	FRACTIONNUMERATOR-SHIFTUP	Standard shift up applied to the numerator.
43	FRACTIONNUMERATOR-DISPLAYSTYLESHIFTUP	Standard shift up applied to the numerator in display style. Suggested: StackTopDisplayStyleShiftUp.
44	FRACTIONDENOMINATOR-SHIFTDOWN	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



\fontdimen	Dimension name	Description
47	FRACTIONNUMDISPLAY- STYLEGAPMIN	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	FRACTIONDENOMINATOR- GAPMIN	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.
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	UNDERBAREXTRA- DESCENDER	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.

\fontdimen	Dimension name	Description
60	RADICALDISPLAYSTYLE- VERTICALGAP	Space between the (ink) top of the expression and the bar over it. Suggested: default rule thickness + $\frac{1}{4}$ 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	RADICALKERNBEFORE- DEGREE	Extra horizontal kern before the degree of a radical, if such is present. Suggested: $\frac{5}{18}$ of em.
64	RADICALKERNAFTERDEGREE	Negative kern after the degree of a radical, if such is present. Suggested: $-\frac{10}{18}$ 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{ABΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}
10 \def\upgreek{ooooooooooooooooooooπoooooooooooooooo}
11 \def\itLatin{oooooooooooooooooooooooooooooooo}
12 \def\itlatin{oooooooooooooooooooooooooooooooo}
13 \def\itGreek{oooooooooooooooooooooooooooooooo}
14 \def\itgreek{oooooooooooooooooooooooooooooooo}
15 \def\testmath#1{
16   \makebox[\linewidth][l]{
17     \makebox[0pt][l]{$\csname up#1\endcsname$}
18     \makebox[0pt][l]{$\csname it#1\endcsname$}}
19 \begin{document}
20 \setmathfont[Colour=2255FF99]{Cambria Math}
21 \parindent=0pt
22 \voffset=-1in
23 \hoffset=-1in
24 \setbox0=\vbox{
25   \testmath{Latin}\\
26   \testmath{latin}\\
27   \testmath{Greek}\\
28   \testmath{greek}}
29 \dimen0=\ht0
30 \advance\dimen0\dp0
31 \edef\papersize{papersize=\the\wd0,\the\dimen0}
32 \setbox255=\vbox{\special{\papersize}\box0}
33 \shipout\box255
34 \end{document}
35 </test>

```

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

## C The bold alphabets

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

```

36 <*testbf>
37 \documentclass{article}
38 \usepackage[a6paper]{geometry}
39 \usepackage{fontspec}
40 \setmainfont{FPL Neu}
41 \usepackage{unicode-math}
42 \def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
43 \def\uplatin{abcdefghijklmnopqrstuvwxyz}
44 \def\upGreek{ABΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}

```



# 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
\'	1167
\:::	665
\::f	665
\::n	665
\@cclvi	302
\@cdots	1190
\@elt	548–552, 555, 559, 561
\@empty	353, 354, 391, 476, 593, 600, 615, 636, 641
\@gobble	1112
\@ifnextchar	1119
\@ifpackageloaded	1193
\@ii	599, 600, 602, 604, 607, 612
\@input	408, 1117
\@marker	612, 631
\@nil	311, 457, 612, 625–628, 1101, 1106, 1115
\@preamblecmds	287
\@sqrt	1119
\@tempa	139, 182, 205, 223, 241, 364, 380, 591, 603, 627, 629, 641
\@tempb	139, 140, 182, 183, 205, 206, 223, 224, 241, 242, 591, 592, 630, 631, 636
\@tempswafalse	601
\@tempswatrue	605, 608, 633, 638, 643, 648
\@um@bfliteraltrue	178, 202
\@um@bfupGreekfalse	145, 184
\@um@bfupGreektrue	157, 169, 190, 196
\@um@bfupLatinfalse	148, 187
\@um@bfupLatintrue	160, 172, 193, 199
\@um@bfupgreekfalse	146, 158, 185, 191
\@um@bfupgreektrue	170, 197
\@um@bfuplatinfalse	149, 188
\@um@bfuplatintrue	161, 173, 194, 200
\@um@fontspec@featuretrue	355
\@um@literaltrue	177
\@um@ot@math@true	383
\@um@upGreekfalse	141
\@um@upGreektrue	153, 165
\@um@upLatinfalse	143, 155
\@um@upLatintrue	167
\@um@upNablafalse	150, 209
\@um@upNablatrie	162, 174, 207
\@um@upgreekfalse	142, 154
\@um@upgreektrue	166
\@um@uplatinfalse	144, 156, 168
\@um@uppartialfalse	147, 159, 186, 192, 227
\@um@uppartialtrue	171, 198, 225
\@	10–13, 17, 23–33, 72–74
\^	33, 1100
Numbers	
\0	28
\_	17–20, 23–32
A	
\A	29
\a	30
\addnolimits	<u>554</u>
\advance	30, 77
\alloc@	302
\Alpha	692
\alpha	717
\AtBeginDocument	690, 1189, 1192
\awint	550
B	
\B	903
\beamer@suppressreplacementstrue	1199
\begin	19, 66
\begingroup	308, 558, 831, 1099, 1113
\Beta	693
\beta	718
\bfitGreek	56
\bfitgreek	57
\bfitLatin	54

`\bfitlatin` ..... 55  
`\bfupgreek` ..... 52  
`\bfupgreek` ..... 53  
`\bfuplatin` ..... 50  
`\bfuplatin` ..... 51  
`\bgroup` ..... 1144  
`\bool_if:NTF` ..... 249  
`\bool_new:N` ..... 22  
`\bool_set_false:N` ... 151, 175, 179, 243  
`\bool_set_true:N` ..... 163, 245  
`\box` ..... 32, 33, 79, 80, 1181

## C

`\C` ..... 892, 918  
`\cdots` ..... 1190  
`\char_make_active:N` ..... 1167  
`\char_make_active:n` ..... 309, 1168  
`\char_make_other:N` ..... 1100  
`\chardef` ..... 302  
`\Chi` ..... 714  
`\chi` ..... 739  
`\cirfnint` ..... 550  
`\clist_map_inline:Nn` ..... 533  
`\clist_map_inline:nn` ... 462, 528, 535  
`\clist_map_variable:NNn` ..... 599  
`\clist_map_variable:nNn` . 656, 667, 673  
`\copy` ..... 1179  
`\cs_gset:cpn` ..... 316, 326  
`\cs_gset:Npn` ..... 329, 334  
`\cs_gset:Npx` ..... 339  
`\cs_gset_eq:NN` ..... 1169, 1170  
`\cs_if_exist:cF` ..... 829  
`\cs_new:Nn` ..... 415, 429, 456, 461,  
     466, 469, 666, 672, 681, 684, 687,  
     813, 828, 841, 1123, 1127, 1143, 1148  
`\cs_new:Npn` ..... 858, 873,  
     889, 901, 916, 925, 930, 935, 940,  
     1017, 1037, 1057, 1062, 1067, 1083  
`\cs_set:cpn` ..... 830  
`\cs_set:Nn` ..... 212,  
     230, 248, 475, 527, 532, 752, 763, 822  
`\cs_set:Npn` ..... 665  
`\cs_set_eq:cN` ..... 823  
`\cs_set_eq:NN` .. 396, 397, 403, 404, 1165  
`\cs_set_protected:cpn` ..... 833  
`\cs_to_str:N` ..... 326, 543

`\csname` ..... 17, 18, 61–64, 305, 311,  
     314, 317, 349, 356, 471, 543, 814, 835

## D

`\DeclareDocumentCommand` ..... 351  
`\DeclareMathVersion` ..... 358  
`\DeclareRobustCommand` ..... 1119  
`\DeclareSymbolFont` ..... 406  
`\def` ..... 7–15, 23, 24,  
     28–83, 88–96, 101–108, 113–121,  
     126–137, 287, 288, 291, 302, 304,  
     306, 357, 392, 547, 555, 557, 559,  
     566, 568, 625, 627–630, 692–747,  
     1108, 1114, 1172, 1183–1188, 1190  
`\define@choicekey` .....  
     ..... 138, 182, 205, 223, 241, 591  
`\define@cmdkey` ..... 587–590  
`\define@key` ..... 570  
`\Delta` ..... 695  
`\delta` ..... 720  
`\dimen` ..... 29–31, 76–78  
`\dimexpr` ..... 289, 382, 1175  
`\documentclass` ..... 2, 37  
`\dp` ..... 30, 77, 1177

## E

`\E` ..... 904  
`\e` ..... 912  
`\edef` ..... 31, 78, 364, 400, 626, 627  
`\egroup` ..... 1160  
`\else` ..... 217,  
     235, 294, 297, 323, 327, 332, 337,  
     340, 384, 398, 436, 449, 480–484,  
     488, 491, 496, 502, 514, 524,  
     560, 573, 606, 611, 617, 635, 640,  
     645, 974, 977, 983, 990, 1003, 1194  
`\else:` ..... 826  
`\encodingdefault` ..... 407  
`\end` ..... 34, 81  
`\endcsname` .. 17, 18, 61–64, 305, 311,  
     314, 317, 349, 356, 471, 543, 814, 835  
`\endgroup` ..... 312, 564, 831, 1111, 1118  
`\Epsilon` ..... 696  
`\epsilon` ..... 721  
`\Eta` ..... 698  
`\eta` ..... 723  
`\etex_ifontchar:D` ..... 826

\ExecuteOptionsX .....	285	\g_um_it_epsilon_letter_usv .....	110, 261, 278
\exp_args:Nnff .....	665, 668, 675	\g_um_it_epsilon_symbol_usv .....	109, 260, 277
\exp_args:No .....	529, 536	\g_um_it_epsilon_usv .....	260, 278
\expandafter .....	305, 313, 318, 320, 324, 542, 555, 602, 604, 607, 612, 626, 627, 631, 835	\g_um_it_phi_letter_usv .....	112, 253, 270
\ExplSyntaxOff .....	1203	\g_um_it_phi_symbol_usv .....	111, 252, 269
\ExplSyntaxOn .....	6	\g_um_it_phi_usv .....	252, 270
<b>F</b>		\g_um_it_varepsilon_usv .....	261, 277
\F .....	905	\g_um_it_varphi_usv .....	253, 269
\f@size .....	356	\g_um_primekern_muskip .....	1120, 1121, 1125
\fi .....	180, 203, 210, 221, 228, 239, 246, 299, 300, 321, 342–346, 390, 405, 439, 454, 481–485, 493, 498, 505, 522, 523, 525, 538, 562, 578, 594, 609, 610, 613, 619, 621, 622, 634, 639, 644, 649–653, 979, 986, 993, 1012, 1015, 1200	\g_um_texgreek_bool .....	22, 151, 163, 175, 179, 243, 245, 249
\fi: .....	826	\g_um_up_epsilon_letter_usv .....	84, 259, 276
\fint .....	550	\g_um_up_epsilon_symbol_usv .....	85, 258, 275
\font .....	381	\g_um_up_epsilon_usv .....	258, 276
\fontdimen .....	289, 382, 1174, 1180	\g_um_up_phi_letter_usv .....	86, 251, 268
<b>G</b>		\g_um_up_phi_symbol_usv .....	87, 250, 267
\g .....	913	\g_um_up_phi_usv .....	250, 268
\g@addto@macro .....	542, 616, 618	\g_um_up_varepsilon_usv .....	259, 275
\g_um_bfit_epsilon_letter_usv .....	122, 265, 282	\g_um_up_varphi_usv .....	251, 267
\g_um_bfit_epsilon_symbol_usv .....	123, 264, 281	\Gamma .....	694
\g_um_bfit_epsilon_usv .....	264, 282	\gamma .....	719
\g_um_bfit_phi_letter_usv .....	124, 257, 274	\gdef .....	1101, 1106
\g_um_bfit_phi_symbol_usv .....	125, 256, 273	\ge .....	1186
\g_um_bfit_phi_usv .....	256, 274	\geq .....	1186
\g_um_bfit_varepsilon_usv .....	265, 281	\glb@currsiz .....	352
\g_um_bfit_varphi_usv .....	257, 273	\global .....	310, 313, 330, 331, 335, 336, 341, 593, 1103, 1108
\g_um_bfup_epsilon_letter_usv .....	97, 263, 280	\group_begin: .....	1166
\g_um_bfup_epsilon_symbol_usv .....	98, 262, 279	\group_end: .....	1171
\g_um_bfup_epsilon_usv .....	262, 280	<b>H</b>	
\g_um_bfup_phi_letter_usv .....	99, 255, 272	\H .....	893, 906, 919
\g_um_bfup_phi_symbol_usv .....	100, 254, 271	\h .....	781, 785, 876
\g_um_bfup_phi_usv .....	254, 272	\hbox .....	1173
\g_um_bfup_varepsilon_usv .....	263, 279	\hoffset .....	23, 70
\g_um_bfup_varphi_usv .....	255, 271	\ht .....	29, 76, 1176
		<b>I</b>	
		\I .....	907, 920
		\if@tempswa .....	614
		\if@um@bfliteral .....	15, 440, 486, 944
		\if@um@bfupGreek .....	16, 499, 987
		\if@um@bfupgreek .....	17, 506, 994
		\if@um@bfupLatin .....	18, 489, 975
		\if@um@bfuplatin .....	19, 494, 980

$\backslash$ if@um@fontspec@feature	8, 571	$\backslash$ l_um_primecount_num	1122, 1145, 1149, 1159
$\backslash$ if@um@literal	10, 431, 478	$\backslash$ l_um_script_features_tl	359, 370
$\backslash$ if@um@ot@math@	9	$\backslash$ l_um_script_font_tl	361, 369
$\backslash$ if@um@upGreek	11, 483	$\backslash$ l_um_sscript_features_tl	360, 374
$\backslash$ if@um@upgreek	12, 484	$\backslash$ l_um_sscript_font_tl	362, 373
$\backslash$ if@um@upLatin	13, 481	$\backslash$ Lambda	702
$\backslash$ if@um@uplatin	14, 482	$\backslash$ lambda	727
$\backslash$ if@um@upNabla	20, 213	$\backslash$ le	1185
$\backslash$ if@um@uppartial	21, 231	$\backslash$ left	567
$\backslash$ ifbeamer@suppressreplacements	1194	$\backslash$ left@primitive	567, 568
$\backslash$ ifcase	140, 183, 206, 224, 242, 592	$\backslash$ leq	1185
$\backslash$ ifdim	382	$\backslash$ let	55, 303, 352–354, 394, 395, 401, 402, 567, 593, 1103, 1112
$\backslash$ ifin@	319, 325	$\backslash$ linewidth	16, 60
$\backslash$ ifnum	534, 632, 637, 642, 646, 647	$\backslash$ lowercase	1102, 1107
$\backslash$ ifx	292, 295, 307, 328, 333, 338, 391, 476, 560, 600, 603, 604, 607, 615, 631, 636, 641	$\backslash$ lowint	552
$\backslash$ iiiint	548	<b>M</b>	
$\backslash$ iiint	548		
$\backslash$ iint	548	$\backslash$ M	909
$\backslash$ in@	318, 324	$\backslash$ m@th	1173
$\backslash$ int	548	$\backslash$ makebox	16–18, 60–64
$\backslash$ intBar	550	$\backslash$ mathaccent	338
$\backslash$ intbar	550	$\backslash$ mathalpha	544, 660
$\backslash$ intcap	552	$\backslash$ mathalphabet	58, 61, 62
$\backslash$ intclockwise	549	$\backslash$ mathbb	890–899
$\backslash$ intcup	552	$\backslash$ mathbf	58, 941–943, 945–973, 976, 978, 981, 982, 984, 985, 988, 989, 991, 992, 995–1002, 1004–1011, 1013, 1014
$\backslash$ intlarhk	551	$\backslash$ mathbffrak	1058–1060
$\backslash$ intx	551	$\backslash$ mathbfit	1018–1035
$\backslash$ Iota	700	$\backslash$ mathbfscr	1063–1065
$\backslash$ iota	725	$\backslash$ mathbfsf	1068–1081
$\backslash$ itGreek	13, 48	$\backslash$ mathbfsfit	1084–1097
$\backslash$ itgreek	14, 49	$\backslash$ mathbfup	1038–1055
$\backslash$ itLatin	11, 46	$\backslash$ mathbin	430
$\backslash$ itlatin	12, 47	$\backslash$ mathcal	1187
<b>K</b>		$\backslash$ mathchar@type	315, 329, 331, 334, 336, 339, 341, 349, 470
		$\backslash$ mathclose	333
$\backslash$ Kappa	701	$\backslash$ mathcode	310
$\backslash$ kappa	726	$\backslash$ mathfrak	917–923
$\backslash$ kern	1174, 1180	$\backslash$ mathgroup	302
<b>L</b>		$\backslash$ mathinner	1190
		$\backslash$ mathit	874–887
$\backslash$ L	908	$\backslash$ mathop	307
$\backslash$ l_um_inc_num	674, 676, 677		
$\backslash$ l_um_incr_num	657, 659, 661		
$\backslash$ l_um_input_num	656, 659, 667, 669, 673, 676		



<code>\mathopen</code> . . . . .	328, 568	<code>\mitSigma</code> . . . . .	710
<code>\mathord</code> . . . . .	432–435,	<code>\mitsigma</code> . . . . .	735
437, 438, 441–448, 450–453, 467		<code>\mitTau</code> . . . . .	711
<code>\mathrm</code> . . . . .	1188	<code>\mittau</code> . . . . .	736
<code>\mathscr</code> . . . . .	902–914, 1187	<code>\mitTheta</code> . . . . .	699
<code>\mathsf</code> . . . . .	926–928	<code>\mittheta</code> . . . . .	724
<code>\mathsf{fit}</code> . . . . .	931–933	<code>\mitUpsilon</code> . . . . .	712
<code>\mathtt</code> . . . . .	936–938	<code>\mitupsilon</code> . . . . .	737
<code>\mathup</code> . . . . .	859–871, 1188	<code>\mitvarepsilon</code> . . . . .	742
<code>\mddefault</code> . . . . .	407	<code>\mitvarkappa</code> . . . . .	744
<code>\mitAlpha</code> . . . . .	692	<code>\mitvarphi</code> . . . . .	745
<code>\mitalpha</code> . . . . .	717	<code>\mitvarpi</code> . . . . .	747
<code>\mitBeta</code> . . . . .	693	<code>\mitvarrho</code> . . . . .	746
<code>\mitbeta</code> . . . . .	718	<code>\mitvarsigma</code> . . . . .	734
<code>\mitChi</code> . . . . .	714	<code>\mitvarTheta</code> . . . . .	709
<code>\mitchi</code> . . . . .	739	<code>\mitvartheta</code> . . . . .	743
<code>\mitDelta</code> . . . . .	695	<code>\mitXi</code> . . . . .	705
<code>\mitdelta</code> . . . . .	720	<code>\mitxi</code> . . . . .	730
<code>\mitEpsilon</code> . . . . .	696	<code>\mitZeta</code> . . . . .	697
<code>\mitepsilon</code> . . . . .	721	<code>\mitzeta</code> . . . . .	722
<code>\mitEta</code> . . . . .	698	<code>\mode_if_math:F</code> . . . . .	834
<code>\miteta</code> . . . . .	723	<code>\mskip</code> . . . . .	1125
<code>\mitGamma</code> . . . . .	694	<code>\Mu</code> . . . . .	703
<code>\mitgamma</code> . . . . .	719	<code>\mu</code> . . . . .	728
<code>\mitIota</code> . . . . .	700	<code>\muskip_gset:Nn</code> . . . . .	1121
<code>\mitiota</code> . . . . .	725	<code>\muskip_new:N</code> . . . . .	1120
<code>\mitKappa</code> . . . . .	701		
<code>\mitkappa</code> . . . . .	726		
<code>\mitLambda</code> . . . . .	702		
<code>\mitlambda</code> . . . . .	727		
<code>\mitMu</code> . . . . .	703		
<code>\mitmu</code> . . . . .	728		
<code>\mitNu</code> . . . . .	704		
<code>\mitnu</code> . . . . .	729		
<code>\mitOmega</code> . . . . .	716		
<code>\mitomega</code> . . . . .	741		
<code>\mitOmicron</code> . . . . .	706		
<code>\mitomicron</code> . . . . .	731		
<code>\mitPhi</code> . . . . .	713		
<code>\mitphi</code> . . . . .	738		
<code>\mitPi</code> . . . . .	707		
<code>\mitpi</code> . . . . .	732		
<code>\mitPsi</code> . . . . .	715		
<code>\mitpsi</code> . . . . .	740		
<code>\mitRho</code> . . . . .	708		
<code>\mitrho</code> . . . . .	733		

\numexpr .....	637,	\ProvidesPackage .....	1
642, 646, 647, 659, 661, 669, 676, 677		\Psi .....	715
		\psi .....	740
<b>O</b>			
\o .....	914	<b>Q</b>	
\oiint .....	548	\Q .....	896
\oint .....	548	<b>R</b>	
\ointctrclockwise .....	548	\R .....	897, 910, 921
\oint .....	548	\r@@t .....	<u>1172</u>
\ointctrclockwise .....	549	\raise .....	1175
\Omega .....	716	\relax .....	140,
\omega .....	741	183, 206, 224, 242, 289, 307, 310,	
\Omicron .....	706	315, 326, 328–331, 333–336, 338,	
\omicron .....	731	339, 341, 349, 352, 381, 382, 534,	
\operator@font .....	1183	592, 604, 607, 631, 632, 637, 642,	
\or .....	152,	646, 647, 659, 661, 669, 676, 677, 1178	
164, 176, 189, 195, 201, 208, 226, 244		\removenolimits .....	<u>557</u>
<b>P</b>			
\P .....	895	\RequirePackage .....	3–5
\PackageError .....	25, 574	\Rho .....	708
\PackageInfo .....	27	\rho .....	733
\PackageWarning .....	26	\rightarrow .....	1184
\PackageWarningNoLine .....	1195	\rootbox .....	1179
\papersize .....	31, 32, 78, 79	\rppolint .....	550
\parindent .....	21, 68	<b>S</b>	
\peek_charcode_remove:NTF ..	1150, 1156	\scan_stop: .....	472, 473, 826
\peek_meaning_remove:NTF .....	1153	\scantokens .....	1103, 1108
\Phi .....	713	\scpolint .....	551
\phi .....	738	\scriptscriptstyle .....	295
\Pi .....	707	\scriptstyle .....	292
\pi .....	732	\setbox .....	24, 32, 71, 79, 1173
\pointint .....	551	\setkeys .....	363
\prg_case_int:nnn .....	1128	\setmainfont .....	5, 40
\prg_do_nothing: .....	823	\SetMathCode .....	<u>348</u> , 463, 544, 658
\prg_new_conditional:Nnn .....	825	\setmathfont .....	20, 67, <u>351</u> , 577
\prg_replicate:nn .....	1125	\sf@size .....	368, 372
\prg_return_false: .....	826	\shipout .....	33, 80
\prg_return_true: .....	826	\Sigma .....	710
\prg_stepwise_variable:nnnNn	657, 674	\sigma .....	735
\prime .....	1165	\space .....	835
\primedouble .....	1131	\special .....	32, 79
\primequadruple .....	1137	\sqint .....	551
\primesingle .....	467, 1124, 1125, 1129	\sqrt .....	566, 1119
\primetriples .....	1134	\sqrtsign .....	1119, 1173
\ProcessOptionsX .....	286	\stepcounter .....	399
\protect .....	577	\string ..	311, 314, 316, 317, 425, 626, 627
\providecommand .....	58		

\strip@pt .....	289	\um@nolimits .....	318, <u>547</u> , 555, 563
\sumint .....	549	\um@PackageError .....	25
<b>T</b>			
\Tau .....	711	\um@PackageInfo .....	27, 393, 425
\tau .....	736	\um@PackageWarning .....	26, 385, 819
\testmath .....	15, 25–28, 59, 72–75	\um@parse@range .....	612, <u>628</u>
\tf@size .....	367, 368	\um@parse@term .....	424, 457, <u>598</u>
\the .....	31, 78	\um@Pool .....	23, 24
\Theta .....	699	\um@radicals .....	324, <u>566</u>
\theta .....	724	\um@resolve@greek .....	<u>690</u>
\theum@fam .....	400	\um@scaled@apply .....	<u>291</u> , 1174, 1180
\thinmuskip .....	1121	\um@scanactivedef .....	311, <u>1099</u>
\tl_new:Nn .....		\um@scancharlet .....	<u>1099</u> , 1115
.. 84–87, 97–100, 109–112, 122–125		\um@set@mathsymbol .....	305, <u>306</u>
\tl_set:Nn .....	214–216,	\um@setmathcode .....	490, 492, 495, 497, 500,
218–220, 232–234, 236–238, 359–362		501, 503, 504, 507–513, 515–521,	
\to .....	1184	<u>655</u> , 750, 753–761, 764–771, 774,	
<b>U</b>			
\um@addto@mathmap .....	529, 536, <u>541</u>	777, 780, 781, 784, 785, 788,	
\um@backslash .....	603, 626	789, 792, 793, 796–802, 805–811	
\um@Break .....	<u>23</u>	\um@symfont .....	392, 400,
\um@char@num@range .....	354, 533, 615, 616, 618	406, 421, 426, 463, 471, 529, 536, 660	
\um@char@range .....	353, 391, 476, 593, 596, 599	\um@usv@bbLatin .....	38, 891
\um@def@itGreek .....	483, <u>791</u>	\um@usv@bblatin .....	39, 899
\um@def@itgreek .....	484, <u>804</u>	\um@usv@bbnum .....	37, 890
\um@def@itLatin .....	481, <u>776</u>	\um@usv@bfDigamma .....	89, 956
\um@def@itlatin .....	482, <u>779</u>	\um@usv@bfDigamma .....	96, 964
\um@def@numbers .....	477, <u>749</u>	\um@usv@bffrakLatin .....	62, 1059
\um@def@upGreek .....	483, <u>787</u>	\um@usv@bffraklatin .....	63, 1060
\um@def@upgreek .....	484, <u>795</u>	\um@usv@bfGreek .....	56, 500,
\um@def@upLatin .....	481, <u>773</u>	503, 768, 949, 988, 1025, 1041, 1045	
\um@def@uplatin .....	482, <u>783</u>	\um@usv@bfgreek .....	57, 507,
\um@firstchar .....	602, 627	515, 769, 951, 995, 1026, 1042, 1046	
\um@firstof .....	625–627	\um@usv@bfitGreek .....	
\um@font .....	293, 296,	60, 500, 503, 770, 950, 991, 1021, 1025	
381, 382, 826, 1174, 1176, 1177, 1180		\um@usv@bfitgreek .....	61,
\um@fontdimen@percent .....		507, 515, 771, 952, 1004, 1022, 1026	
..... <u>288</u> , 293, 296, 1176, 1177		\um@usv@bfith .....	114, 953, 985
\um@Loop .....	<u>23</u>	\um@usv@bfitLatin .....	
\um@mathsymbol .....	<u>304</u> , 421, 426	58, 490, 492, 766, 946, 978, 1019, 1023	
\um@mathsymbol@noparse .....	394, 420	\um@usv@bfitlatin .....	
\um@mathsymbol@parse .....	401, <u>423</u>	59, 495, 497, 767, 948, 984, 1020, 1024	
\um@mversion .....	357, 358	\um@usv@bfitNabla .....	
		..... 129, 219, 442, 450, 966, 1028	
		\um@usv@bfitpartial .....	
		. 135, 237, 446, 452, 967, 1011, 1029	

$\backslash\mathrm{um@usv@bfitvarepsilon}$ . . . . .	$\backslash\mathrm{um@usv@bfvarkappa}$ . . . . .
. . . . . 116, 508, 516, 968, 1005, 1030	. . . . . 92, 510, 518, 960, 998, 1052
$\backslash\mathrm{um@usv@bfitvarkappa}$ . . . . .	$\backslash\mathrm{um@usv@bfvarphi}$ . . . . .
. . . . . 118, 510, 518, 970, 1007, 1032	. . . . . 93, 511, 519, 961, 999, 1053
$\backslash\mathrm{um@usv@bfitvarphi}$ . . . . .	$\backslash\mathrm{um@usv@bfvarpi}$ . . . . .
. . . . . 119, 511, 519, 971, 1008, 1033	. . . . . 95, 513, 521, 963, 1001, 1055
$\backslash\mathrm{um@usv@bfitvarpi}$ . . . . .	$\backslash\mathrm{um@usv@bfvarrho}$ . . . . .
. . . . . 121, 513, 521, 973, 1010, 1035	. . . . . 94, 512, 520, 962, 1000, 1054
$\backslash\mathrm{um@usv@bfitvarrho}$ . . . . .	$\backslash\mathrm{um@usv@bfvarTheta}$ . . . . .
. . . . . 120, 512, 520, 972, 1009, 1034	. . . . . 88, 501, 504, 954, 989, 1047
$\backslash\mathrm{um@usv@bfitvarTheta}$ . . . . .	$\backslash\mathrm{um@usv@bfvartheta}$ . . . . .
. . . . . 115, 501, 504, 965, 992, 1027	. . . . . 91, 509, 517, 959, 997, 1051
$\backslash\mathrm{um@usv@bfitvartheta}$ . . . . .	$\backslash\mathrm{um@usv@Digamma}$ . . . . .
. . . . . 117, 509, 517, 969, 1006, 1031	. . . . . 76, 942, 956
$\backslash\mathrm{um@usv@bfLatin}$ . . . . . 53, 490,	$\backslash\mathrm{um@usv@digamma}$ . . . . . 83, 943, 964
492, 764, 945, 976, 1023, 1039, 1043	$\backslash\mathrm{um@usv@frakLatin}$ . . . . . 42, 917, 1059
$\backslash\mathrm{um@usv@bflatin}$ . . . . . 54, 55, 495,	$\backslash\mathrm{um@usv@fraklatin}$ . . . . . 43, 923, 1060
497, 765, 947, 981, 1024, 1040, 1044	$\backslash\mathrm{um@usv@itGreek}$ . . . . .
$\backslash\mathrm{um@usv@bfNabla}$ . . . . .	. . . . . 35, 760, 788, 792, 861, 877,
. . . . . 128, 215, 441, 450, 955, 1048	950, 988, 991, 1021, 1041, 1071, 1087
$\backslash\mathrm{um@usv@bfnum}$ . . . . . 52, 941,	$\backslash\mathrm{um@usv@itgreek}$ 36, 796, 805, 862, 878,
1018, 1038, 1058, 1063, 1068, 1084	952, 995, 1004, 1022, 1042, 1072, 1088
$\backslash\mathrm{um@usv@bfpartial}$ . . . . .	$\backslash\mathrm{um@usv@ith}$ . . . . .
. . . . . 134, 233, 445, 452, 957, 1002, 1049	101, 756, 781, 785, 876, 953, 982, 985
$\backslash\mathrm{um@usv@bfscrLatin}$ . . . . . 64, 1064	$\backslash\mathrm{um@usv@itLatin}$ . . . . . 31,
$\backslash\mathrm{um@usv@bfscrLatin}$ . . . . . 65, 1065	754, 774, 777, 859, 874, 891, 902,
$\backslash\mathrm{um@usv@bfsfGreek}$ . . . . . 69, 1071	917, 927, 932, 937, 946, 976, 978,
$\backslash\mathrm{um@usv@bfsfGreek}$ . . . . . 70, 1072	1019, 1039, 1059, 1064, 1069, 1085
$\backslash\mathrm{um@usv@bfsfitGreek}$ . . . . . 73, 1087	$\backslash\mathrm{um@usv@itlatin}$ . . . . . 32,
$\backslash\mathrm{um@usv@bfsfitGreek}$ . . . . . 74, 1088	755, 780, 784, 860, 875, 899, 911,
$\backslash\mathrm{um@usv@bfsfitLatin}$ . . . . . 71, 1085	923, 928, 933, 938, 948, 981, 984,
$\backslash\mathrm{um@usv@bfsfitLatin}$ . . . . . 72, 1086	1020, 1040, 1060, 1065, 1070, 1086
$\backslash\mathrm{um@usv@bfsfitNabla}$ . . . . .	$\backslash\mathrm{um@usv@itNabla}$ 127, 218, 433, 437, 863,
. . . . . 131, 220, 444, 451, 1090	879, 966, 1013, 1028, 1048, 1074, 1090
$\backslash\mathrm{um@usv@bfsfitpartial}$ . . . . .	$\backslash\mathrm{um@usv@itpartial}$ . . . . . 133,
. . . . . 137, 238, 448, 453, 1091	236, 435, 438, 864, 880, 967, 1002,
$\backslash\mathrm{um@usv@bfsfLatin}$ . . . . . 67, 1069	1011, 1014, 1029, 1049, 1075, 1091
$\backslash\mathrm{um@usv@bfsflatIn}$ . . . . . 68, 1070	$\backslash\mathrm{um@usv@itvarepsilon}$ . . . . .
$\backslash\mathrm{um@usv@bfsfNabla}$ . . . . . 130, 216, 443, 451	. . . . . 103, 797, 806, 866, 882,
$\backslash\mathrm{um@usv@bfsfnum}$ . . . . . 66	968, 996, 1005, 1030, 1050, 1076, 1092
$\backslash\mathrm{um@usv@bfsfpartial}$ . . . . . 136, 234, 447, 453	$\backslash\mathrm{um@usv@itvarkappa}$ . . . . .
$\backslash\mathrm{um@usv@bfuph}$ . . . . . 113, 982	. . . . . 105, 799, 808, 868, 884,
$\backslash\mathrm{um@usv@bfuplatin}$ . . . . . 55	970, 998, 1007, 1032, 1052, 1078, 1094
$\backslash\mathrm{um@usv@bfvarepsilon}$ . . . . .	$\backslash\mathrm{um@usv@itvarphi}$ . . . . .
. . . . . 90, 508, 516, 958, 996, 1050	. . . . . 106, 800, 809, 869, 885,
	971, 999, 1008, 1033, 1053, 1079, 1095
	$\backslash\mathrm{um@usv@itvarpi}$ . . . . .

...	108, 802, 811, 871, 887, 973,
1001, 1010, 1035, 1055, 1081, 1097	
\um@usv@itvarrho	.....
...	107, 801, 810, 870, 886, 972,
1000, 1009, 1034, 1054, 1080, 1096	
\um@usv@itvarTheta	... 102, 793, 865,
881, 965, 989, 992, 1027, 1047, 1073	
\um@usv@itvartheta	.....
.....	104, 798, 807, 867, 883,
969, 997, 1006, 1031, 1051, 1077, 1093	
\um@usv@Nabla	126, 214, 432, 437, 863,
879, 955, 1013, 1028, 1048, 1074, 1090	
\um@usv@num	.....
.	28, 750, 890, 926, 931, 936, 941,
1018, 1038, 1058, 1063, 1068, 1084	
\um@usv@partial	..... 132,
232, 434, 438, 864, 880, 957, 1002,	
1011, 1014, 1029, 1049, 1075, 1091	
\um@usv@scrLatin	..... 40, 902
\um@usv@scrLatin	..... 41, 911
\um@usv@sfitLatin	..... 47, 932
\um@usv@sfitLatin	..... 48, 933
\um@usv@sflatin	..... 45, 927
\um@usv@sflatin	..... 46, 928
\um@usv@sfnun	..... 44, 926, 931
\um@usv@ttLatin	..... 50, 937
\um@usv@ttLatin	..... 51, 938
\um@usv@ttnum	..... 49, 936
\um@usv@upGreek	.....
...	33, 758, 788, 792, 861, 877,
949, 988, 991, 1021, 1041, 1071, 1087	
\um@usv@upgreek	.....
...	34, 761, 796, 805, 862, 878,
951, 995, 1004, 1022, 1042, 1072, 1088	
\um@usv@upLatin	..... 29,
753, 774, 777, 859, 874, 891, 902,	
917, 927, 932, 937, 945, 976, 978,	
1019, 1039, 1059, 1064, 1069, 1085	
\um@usv@uplatin	..... 30,
757, 780, 784, 860, 875, 899, 911,	
923, 928, 933, 938, 947, 981, 984,	
1020, 1040, 1060, 1065, 1070, 1086	
\um@usv@varepsilon	.....
.....	77, 797, 806, 866, 882,
958, 996, 1005, 1030, 1050, 1076, 1092	
\um@usv@varkappa	.....
.....	79, 799, 808, 868, 884,
960, 998, 1007, 1032, 1052, 1078, 1094	
\um@usv@varphi	80, 800, 809, 869, 885,
961, 999, 1008, 1033, 1053, 1079, 1095	
\um@usv@varpi	.....
...	82, 802, 811, 871, 887, 963,
1001, 1010, 1035, 1055, 1081, 1097	
\um@usv@varrho	.....
...	81, 801, 810, 870, 886, 962,
1000, 1009, 1034, 1054, 1080, 1096	
\um@usv@varTheta	.....
...	75, 759, 789, 793, 865, 881,
954, 989, 992, 1027, 1047, 1073, 1089	
\um@usv@vartheta	.....
.....	78, 798, 807, 867, 883,
959, 997, 1006, 1031, 1051, 1077, 1093	
\um@zf@feature	..... 569, 581, 584
\um@bfNabla_up_or_it_usv	.....
.....	215, 219, 450, 1013
\um@bfpartial_up_or_it_usv	.....
.....	233, 237, 452, 1014
\um@bfsfNabla_up_or_it_usv	216, 220, 451
\um@bfsfpartial_up_or_it_usv	...
.....	234, 238, 453
\um@config@mathbb:	..... 889
\um@config@mathbf:	..... 940
\um@config@mathbffrac:	..... 1057
\um@config@mathbfrit:	..... 1017
\um@config@mathbfscr:	..... 1062
\um@config@mathbfssf:	..... 1067
\um@config@mathbfssrit:	..... 1083
\um@config@mathbfup:	..... 1037
\um@config@mathfrak:	..... 916
\um@config@mathit:	..... 873
\um@config@mathscr:	..... 901
\um@config@mathsf:	..... 925
\um@config@mathsfit:	..... 930
\um@config@mathtt:	..... 935
\um@config@mathup:	..... 858
\um@glyph_if_exist:n	..... 825
\um@glyph_if_exist:nTF	.....
.....	814, 825, 1131, 1134, 1137
\um@init@alphabet:n	..... 397, 822
\um@make@mathactive:nNN	.... 467, 469
\um@mathmap:Nnn	..... 395, 402, 668, 675

<code>\um_mathmap_noparse:Nnn</code> . . . . .	395, <a href="#">527</a>	<code>\um_setup_alphanum:</code> . . . . .	412, <a href="#">475</a>
<code>\um_mathmap_parse:Nnn</code> . . . . .	402, <a href="#">532</a>	<code>\um_setup_bf_literals:</code> . . . . .	487, <a href="#">763</a>
<code>\um_maybe_init_alphabet:n</code> 397, 404, 815		<code>\um_setup_literals:</code> . . . . .	479, <a href="#">752</a>
<code>\um_Nabla_up_or_it_usv</code> . . 214, 218, 437		<code>\um_setup_math_alphabet:n</code> 813, 842–856	
<code>\um_nprimes:n</code> 1123, 1131, 1134, 1137, 1140		<code>\um_setup_mathactives:</code> . . . . .	411, 466
<code>\um_nprimes_select:n</code> . . . . .	1127, 1159	<code>\um_setup_mathup:</code> . . . . .	1183
<code>\um_partial_up_or_it_usv</code> 232, 236, 438		<code>\um_setup_nabla:</code> . . . . .	212, 416
<code>\um_prepare_alpha:n</code> . . . . .	816, <a href="#">828</a>	<code>\um_setup_partial:</code> . . . . .	230, 417
<code>\um_remap_symbol:nnn</code> . 396, 403, 430,		<code>\um_setup_shapes:</code> . . . . .	409, 415
432–435, 437, 438, 441–448, 450–453		<code>\um_setup_vargreek:</code> . . . . .	248, 418
<code>\um_remap_symbol_noparse:nnn</code> . . . . .		<code>\unicodemathgobble</code> . . . . .	1112
. . . . .	396, 458, 461	<code>\UnicodeMathSymbol</code> . . . . .	394, 401, 1114
<code>\um_remap_symbol_parse:nnn</code> . . 403, 456		<code>\unless</code> . . . . .	600
<code>\um_remap_symbols:</code> . . . . .	410, <a href="#">429</a>	<code>\updefault</code> . . . . .	407
<code>\um_scanprime:</code> . . . . .		<code>\upGreek</code> . . . . .	9, 44
. . . . .	1143, 1153, 1165, 1169, 1170	<code>\upgreek</code> . . . . .	10, 45
<code>\um_scanprime_collect:</code> . . . . .		<code>\upint</code> . . . . .	552
. . . . .	1146, 1148, 1151, 1154, 1157	<code>\upLatin</code> . . . . .	7, 42
<code>\um_set_mathalph_range:Nnn</code> . . . . .	<a href="#">672</a>	<code>\uplatin</code> . . . . .	8, 43
<code>\um_set_mathalph_range:nNnn</code> . . . . .		<code>\Upsilon</code> . . . . .	712
. . . . .	672, 682, 685, 688	<code>\Upsilon</code> . . . . .	737
<code>\um_set_mathalphabet_char:Nnn</code> 666,		<code>\use:c</code> . . . . .	817, 831, 837
863–871, 876, 879–887, 892–898,		<code>\use_none:n</code> . . . . .	404
903–910, 912–914, 918–922,		<code>\usepackage</code> . . . . .	3, 4, 6, 38, 39, 41
942, 943, 953–973, 982, 985,			
989, 992, 996–1002, 1005–1011,			
1013, 1014, 1027–1035,			
1047–1055, 1073–1081, 1089–1097			
<code>\um_set_mathalphabet_char:Nnnn</code> . . <a href="#">665</a>			
<code>\um_set_mathalphabet_greek:Nnn</code> . .			
687, 861, 862, 877, 878, 949–952,			
988, 991, 995, 1004, 1021,			
1022, 1025, 1026, 1041, 1042,			
1045, 1046, 1071, 1072, 1087, 1088			
<code>\um_set_mathalphabet_latin:Nnn</code> . .			
. . . . .	684,		
859, 860, 874, 875, 891, 899, 902,			
911, 917, 923, 927, 928, 932, 933,			
937, 938, 945–948, 976, 978, 981,			
984, 1019, 1020, 1023, 1024, 1039,			
1040, 1043, 1044, 1059, 1060,			
1064, 1065, 1069, 1070, 1085, 1086			
<code>\um_set_mathalphabet_numbers:Nnn</code> .			
. . . 681, 890, 926, 931, 936, 941,			
1018, 1038, 1058, 1063, 1068, 1084			
<code>\um_setup_alphabets:</code> . . . . .	413, 841		

<code>\XeTeXmathcode</code> . . . . .	331, 336, 341, 349		<b>Z</b>
<code>\XeTeXmathcodenum</code> . . . . .	473	<code>\Z</code> . . . . .	898, 922
<code>\XeTeXradical</code> . . . . .	326	<code>\z@</code> . . . . .	1173, 1176, 1177, 1181
<code>\Xi</code> . . . . .	705	<code>\Zeta</code> . . . . .	697
<code>\xi</code> . . . . .	730	<code>\zeta</code> . . . . .	722
<code>\XKV@rm</code> . . . . .	377	<code>\zf@family</code> . . . . .	407
		<code>\zf@fontspec</code> . . . . .	364
		<code>\zf@update@ff</code> . . . . .	582, 585