

Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/09/29 v0.4

Abstract

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

Contents

1	Introduction	2			
2	Unicode maths font setup	2			
2.1	Using multiple fonts	3			
2.2	Script and scriptscript fonts/features	3			
3	Maths input	3			
3.1	Math ‘style’	4			
3.2	Bold style	5			
3.3	Sans serif style	5			
3.4	All (the rest) of the mathematical alphabets	6			
3.5	Miscellanea	6			
I	The unicode-math package	12			
4	Things we need	13			
4.1	Package options	16			
4.2	Overcoming \@on-lypreamble	19			
	4.3 Other things	20			
5	Fundamentals	21			
5.1	Enlarging the number of maths families	21			
5.2	\DeclareMathSymbol for unicode ranges	21			
5.3	The main \setmathfont macro	23			
5.4	(Big) operators	30			
5.5	Radicals	34			
5.6	Delimiters	34			
5.7	Maths accents	36			
6	Font features	37			
6.1	OpenType maths font features	38			
6.2	Script and scriptscript font options	38			
6.3	Range processing	38			
6.4	Resolving Greek symbol name control sequences	42			

6.5	Setting up the mappings	44	A Documenting maths support in the NFSS	69
7	Maths alphabets mapping definitions	46	A.1 Overview	69
7.1	Non-bold math alphabets	49	III X_YTeX math font dimensions	70
7.2	Bold math alphabets	52		
7.3	Definitions of the math symbols	58	IV Some manner of unit testing	75
8	Epilogue	60	B The bold alphabets	75
II	stix table data extraction	67		

1 Introduction

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

Users who desire to specify maths alphabets only from various fonts may wish to use Andrew Moschou’s mathspec package instead.

2 Unicode maths font setup

In the ideal case, a single unicode font will contain all maths glyphs we need. The file unicode-math-table.tex (based on 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}}
```

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

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

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

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

This has not been implemented yet.

Instances above of

`[]{}`

follow from my fontspec package, and therefore any additional ** 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:

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

where *<unicode range>* is a comma-separated list of unicode slots and ranges such as `{27D0–27EB, 27FF, 295B–297F}`. You may also use the macro for accessing the glyph, such as `\`, or whole collection of symbols with the same math type, such as `\mathopen`. (Only numerical slots, however, can be used in proper ranges.) This interface still requires some thought.

Not yet implemented: preset names ranges could be used in the range spec., 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.

Not yet implemented: Both of these options must be taken into account. I hope this will be mostly automatic from the users' points of view. The `+ssty` feature can be detected and applied automatically, and appropriate optical size information embedded in the fonts will ensure this latter case. Fine tuning should be possible automatically with fontspec options. We might have to wait until `MnMath`, for example, before we really know.

3 Maths input

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

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

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

3.1 Math ‘style’

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

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

The philosophy behind the interface to the mathematical alphabet symbols lies in \LaTeX ’s attempt of separating content and formatting. Because input source text may come from a variety of places, the upright and ‘mathematical’ italic Latin and Greek alphabets are *unified* from the point of view of having a specified meaning in the source text. That is, to get a mathematical ‘ x ’, either the ascii (‘keyboard’) letter `x` may be typed, or the actual unicode character may be used. Similarly for Greek letters. The upright or italic forms are then chosen based on the `math-style` package option.

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

Alternative interface However, some users may not like this convention. For them, an upright `x` is an upright ‘ x ’ and that’s that. (This will be the case when obtaining source text from copy/pasting PDF or Microsoft Word documents, for example.) For these users, the `literal` option to `math-style` will effect this behaviour.

The `math-style` options’ effects are shown in brief in table 1.

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

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

3.2 Bold style

Similar as in the previous section, ISO standards differ somewhat to \TeX 's conventions (and classical typesetting) for 'boldness' in mathematics. In the past, it has been customary to use bold *upright* letters to denote things like vectors and matrices. For example, $\mathbf{M} = (M_x, M_y, M_z)$. Presumably, this was due to the relatively scarcity of bold italic fonts in the pre-digital typesetting era. It has been suggested that *italic* bold symbols are used nowadays instead.

Bold Greek letters have simply been bold variant glyphs of their regular weight, as in $\boldsymbol{\xi} = (\xi_r, \xi_\varphi, \xi_\theta)$. Confusingly, the syntax in \LaTeX has been different for these two examples: `\mathbf{b}` in the former (\mathbf{M}), and `\bm` (or `\boldsymbol`, deprecated) in the latter ($\boldsymbol{\xi}$).

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

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

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

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

The `bold-style` options' effects are shown in brief in table 2.

3.3 Sans serif style

Unicode contains upright and italic, medium and bold mathematical alphabet characters. These may be explicitly selected with the `\mathsfup`, `\mathsfit`, `\mathbfsup`, and `\mathbfsfit` commands discussed in section §3.4.

How should the generic `\mathsf` behave? Unlike bold, sans serif is used much more sparingly in mathematics. I've seen recommendations to typeset tensors in

sans serif italic or sans serif italic bold (e.g., examples in the `isomath` and `mattens` packages). But L^AT_EX's `\mathsf` is *upright* sans serif.

Therefore I reluctantly add the package options `[sans-style=TeX]` and `[sans-style=ISO]` to control the behaviour of `\mathsf`. The `TeX` style sets up the command to use the seemingly-useless upright sans serif, including Greek; the `ISO` style switches to using italic in both Latin and Greek alphabets. In other words, this option simply changes the meaning of `\mathsf` to either `\mathsfup` or `\mathsf-fit`, respectively. Please let me know if more granular control is necessary here.

There is also a `[sans-style=literal]` setting, set automatically with `[math-style=literal]`, which retains the uprightness of the input characters used when selecting the sans serif output.

3.3.1 What about bold sans serif?

While you might want your bold upright and your sans serif italic, I don't believe you'd also want your bold sans serif upright (or all vice versa, if that's even conceivable). Therefore, bold sans serif follows from the setting for sans serif; it is completely independent of the setting for bold.

In other words, `\mathbfsf` is `\mathbfsfup` or `\mathbfsffit` based on `[sans-style=TeX]` or `[sans-style=ISO]`, respectively. And `[sans-style=literal]` causes `\mathbfsf` to retain the same italic or upright shape as the input, and turns it bold sans serif.

Note well! There is no medium-weight sans serif Greek alphabet in unicode; therefore, `\mathsf{\alpha}` does not make sense (simply produces 'α') while `\mathbfsf{\alpha}` gives 'α'.

3.4 All (the rest) of the mathematical alphabets

Unicode contains separate codepoints for most if not all variations of alphabet shape one may wish to use in mathematical notation. The complete list is shown in table 3. Some of these have been covered in the previous sections.

At present, the math font switching commands do not nest; therefore if you want sans serif bold, you must write `\mathsfbf{...}` rather than `\mathbf{\mathsf{...}}`. This may change in the future.

3.5 Miscellanea

3.5.1 Nabla

The symbol ∇ comes in the six forms shown in table 4. 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). T_EX classically uses an upright nabla, but iso standards differ (I think). The package options `nabla=upright` and

Table 3: Mathematical alphabets defined in unicode. Black dots indicate an alphabet exists in the font specified; grey dots indicate shapes that should always be taken from the upright font even in the italic style.

Font				Alphabet		
Style	Shape	Series	Switch	Latin	Greek	Numerals
Serif	Upright	Normal	<code>\mathup</code>	•	•	•
		Bold	<code>\mathbfup</code>	•	•	•
	Italic	Normal	<code>\mathit</code>	•	•	•
		Bold	<code>\mathbfit</code>	•	•	•
Sans serif	Upright	Normal	<code>\mathsfup</code>	•		•
	Italic	Normal	<code>\mathsfit</code>	•		•
	Upright	Bold	<code>\mathsfbfup</code>	•	•	•
	Italic	Bold	<code>\mathsfbfit</code>	•	•	•
Typewriter	Upright	Normal	<code>\mathtt</code>	•		•
Double-struck	Upright	Normal	<code>\mathbb</code>	•		•
Script	Upright	Normal	<code>\mathscr</code>	•		
		Bold	<code>\matbfscr</code>	•		
Fraktur	Upright	Normal	<code>\mathfrak</code>	•		
		Bold	<code>\mathbffrac</code>	•		

`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`.

3.5.2 Partial

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

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

Use the `partial=upright` or `partial=italic` package options to specify which one you would like. The default is (always, unless someone requests and argues otherwise) `partial=italic`.¹

See table 5 for the variations on the partial differential symbol.

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

Table 4: The various forms of nabla.

Description		Glyph
Upright	Serif	∇
	Bold serif	∇
	Bold sans	∇
Italic	Serif	<i>∇</i>
	Bold serif	<i>∇</i>
	Bold sans	<i>∇</i>

Table 5: The various forms of the partial differential. Note that in the fonts used to display these glyphs, the first upright partial is incorrectly shown in an italic style.

Description		Glyph
Regular	Upright	∂
	Italic	<i>∂</i>
Bold	Upright	∂
	Italic	<i>∂</i>
Sans bold	Upright	∂
	Italic	<i>∂</i>

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

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

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

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

U+3B5: GREEK SMALL LETTER EPSILON

U+3F5: GREEK LUNATE EPSILON SYMBOL

U+3C6: GREEK SMALL LETTER PHI

U+3D5: GREEK SMALL LETTER SCRIPT PHI

3.5.4 Primes

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

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

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

3.5.5 Unicode subscripts and superscripts

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

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

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

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

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

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

3.5.6 Colon ‘:’

The colon is one of the few confusing characters of unicode maths. In \TeX , `:` is defined as a colon with relation spacing: ‘ $a : b$ ’. While `\colon` is defined as a colon with punctuation spacing: ‘ $a : b$ ’.

In unicode, `U+003A: COLON` is defined as a punctuation symbol, while `U+2236: RATIO` is the colon-like symbol used in mathematics to denote ratios and other things.

This breaks the usual straightforward mapping from control sequence to unicode input character to (the same) unicode glyph.

To preserve input compatibility, we remap the `ASCII` input character ‘:’ to `U+2236: RATIO`. Typing a literal `U+2236: RATIO` char will result in the same output. If `amsmath` is loaded, then the definition of `\colon` is inherited from there (it looks like a punctuation colon with additional space around it). Otherwise, `\colon` is made to output a colon with `\mathpunct` spacing.

The package option `[colon=literal]` forces `ASCII` input ‘:’ to be printed as `\mathcolon` instead.

3.5.7 Slashes and backslashes

There are several slash-like symbols defined in unicode. These are shown in table 6. The `ASCII` slashes `/` and `\` are useful as input characters but should not be used in the rendering of mathematics. (I think.)

In regular \LaTeX we can write `\left\slash...\right\backslash` and so on and obtain extensible delimiter-like symbols. Not all of the unicode slashes are suitable for this (and do not have the font support to do it).

Table 6: Slashes and backslashes.

Slot	Name	Glyph	Command
U+002F	SOLIDUS	/	\solidus
U+2044	FRACTION SLASH	/	\fracslash
U+2215	DIVISION SLASH	/	\slash
U+29F8	BIG SOLIDUS	/	\xsol
U+005C	REVERSE SOLIDUS	\	\backslash
U+2216	SET MINUS	\	\smallsetminus
U+29F5	REVERSE SOLIDUS OPERATOR	\	\setminus
U+29F9	BIG REVERSE SOLIDUS	\	\xbsol

Slash Of U+2044: FRACTION SLASH, TR25 says that it is:

...used to build up simple fractions in running text...however parsers of mathematical texts should be prepared to handle fraction slash when it is received from other sources.

If encountered in the input stream, therefore, I believe it should be mapped to the meaning of U+2215: DIVISION SLASH. (Alas, see the note below.)

U+2215: DIVISION SLASH should be used when division is represented without a built-up fraction; $\pi \approx 22/7$, for example.

I do not know what U+29F8: BIG SOLIDUS is intended to be used for. It's a 'math operator' (like Σ) so it falls outside the topic of discussion here.

Backslash MathML uses U+2216: SET MINUS like this: $A \setminus B$.² I think the STIX name for this glyph slot should just be \setminus.

Presumably, U+29F5: REVERSE SOLIDUS OPERATOR is intended to be used in a similar way, but it could also (perhaps?) be used to represent 'inverse division': $\pi \approx 7 \setminus 22$.³

Again, I don't know what U+29F9: BIG REVERSE SOLIDUS is for. But it's not too important at this stage.

How to use all of these things Unfortunately, font support for the above characters/glyphs is rather spotty. In Cambria Math, the only slash that grows (say when writing

$$\left[\begin{array}{cc} a & b \\ c & d \end{array} \right] / \left[\begin{array}{cc} 1 & 1 \\ 1 & 0 \end{array} \right])$$

²§4.4.5.11 :// . 3. / / 3/

³This is valid syntax in the Octave and Matlab programming languages, in which it means matrix inverse pre-multiplication. I.e., $A \setminus B = A^{-1}B$.

is the `FRACTION SLASH`, which we just established above is sort of only supposed to be used in text.

And none of the backslashes stretch. Which leaves me in a bit of a pickle. \TeX has a stretchy backslash. Cambria Math does not. What will? And in which glyph slot? I give up, for now. This is an impossible problem.

All of the above characters are allowed to be used after `\left`, `\middle`, and `\right`. Only the font will know whether or not it will actually stretch, however. If you like you may redefine `\slash` and `\backslash` to fit your needs. Perhaps this will be a package option some day.

3.5.8 Normalising some input characters

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

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

(Not yet implemented.)

File I

The unicode-math package

This is the package.

```
1 \ProvidesPackage{unicode-math}  
2 [2009/09/29 v0.4 Unicode maths in XeLaTeX]
```

4 Things we need

Packages

```
3 \RequirePackage{expl3}[2009/08/12]
4 \RequirePackage{xparse}[2009/08/31]
5 \RequirePackage{fontspec}
   Start using LATEX3 — 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
```

4.0.9 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.⁴

```
23 \def\um@usv@num{'\0}
24 \def\um@usv@upLatin{'\A}
25 \def\um@usv@uplatin{'\a}
26 \def\um@usv@upGreek{"391}
27 \def\um@usv@upgreek{"3B1}
28 \def\um@usv@itLatin{"1D434}
29 \def\um@usv@itlatin{"1D44E}
```

⁴'u.s.v.' stands for 'unicode scalar value'.

```

30 \def\um@usv@itGreek{"1D6E2}
31 \def\um@usv@itgreek{"1D6FC}
32 \def\um@usv@bbnum{"1D7D8}
33 \def\um@usv@bbLatin{"1D538}
34 \def\um@usv@bbLatin{"1D552}
35 \def\um@usv@scrLatin{"1D49C}
36 \def\um@usv@scrLatin{"1D4B6}
37 \def\um@usv@frakLatin{"1D504}
38 \def\um@usv@frakLatin{"1D51E}
39 \def\um@usv@sfnum{"1D7E2}
40 \def\um@usv@sfupLatin{"1D5A0}
41 \def\um@usv@sfLatin {"1D5A0}
42 \def\um@usv@sfupLatin{"1D5BA}
43 \def\um@usv@sfitLatin{"1D608}
44 \def\um@usv@sfitLatin{"1D622}
45 \def\um@usv@ttnum{"1D7F6}
46 \def\um@usv@ttLatin{"1D670}
47 \def\um@usv@ttLatin{"1D68A}

```

Bold:

```

48 \def\um@usv@bfnum{"1D7CE}
49 \def\um@usv@bfupLatin{"1D400}
50 \def\um@usv@bfLatin {"1D400}
51 \def\um@usv@bfupLatin{"1D41A}
52 \def\um@usv@bfupGreek{"1D6A8}
53 \def\um@usv@bfupGreek{"1D6C2}
54 \def\um@usv@bfitLatin{"1D468}
55 \def\um@usv@bfitLatin{"1D482}
56 \def\um@usv@bfitGreek{"1D71C}
57 \def\um@usv@bfitGreek{"1D736}
58 \def\um@usv@bffrakLatin{"1D56C}
59 \def\um@usv@bffrakLatin{"1D586}
60 \def\um@usv@bfscrLatin{"1D4D0}
61 \def\um@usv@bfscrLatin{"1D4EA}
62 \def\um@usv@bfsfnum{"1D7EC}
63 \def\um@usv@bfsfupLatin{"1D5D4}
64 \def\um@usv@bfsfLatin {"1D5D4}
65 \def\um@usv@bfsfupLatin{"1D5EE}
66 \def\um@usv@bfsfupGreek{"1D756}
67 \def\um@usv@bfsfupGreek{"1D770}
68 \def\um@usv@bfsfitLatin{"1D63C}
69 \def\um@usv@bfsfitLatin{"1D656}
70 \def\um@usv@bfsfitGreek{"1D790}
71 \def\um@usv@bfsfitGreek{"1D7AA}

```

Greek variants:

```

72 \def\um@usv@varTheta{"3F4}
73 \def\um@usv@Digamma{"3DC}

```

```

74 \def\um@usv@varepsilon{"3F5}
75 \def\um@usv@vartheta{"3D1}
76 \def\um@usv@varkappa{"3F0}
77 \def\um@usv@varphi{"3D5}
78 \def\um@usv@varrho{"3F1}
79 \def\um@usv@varpi{"3D6}
80 \def\um@usv@digamma{"3DD}

```

Bold:

```

81 \def\um@usv@bfvarTheta{"1D6B9}
82 \def\um@usv@bfDigamma{"1D7CA}
83 \def\um@usv@bfvarepsilon{"1D6DC}
84 \def\um@usv@bfvartheta{"1D6DD}
85 \def\um@usv@bfvarkappa{"1D6DE}
86 \def\um@usv@bfvarphi{"1D6DF}
87 \def\um@usv@bfvarrho{"1D6E0}
88 \def\um@usv@bfvarpi{"1D6E1}
89 \def\um@usv@bfdigamma{"1D7CB}

```

Italic Greek variants:

```

90 \def\um@usv@ith{"210E}
91 \def\um@usv@itvarTheta{"1D6F3}
92 \def\um@usv@itvarepsilon{"1D716}
93 \def\um@usv@itvartheta{"1D717}
94 \def\um@usv@itvarkappa{"1D718}
95 \def\um@usv@itvarphi{"1D719}
96 \def\um@usv@itvarrho{"1D71A}
97 \def\um@usv@itvarpi{"1D71B}

```

Bold:

```

98 \def\um@usv@bfuph{"1D421}
99 \def\um@usv@bfith{"1D489}
100 \def\um@usv@bfitvarTheta{"1D72D}
101 \def\um@usv@bfitvarepsilon{"1D750}
102 \def\um@usv@bfitvartheta{"1D751}
103 \def\um@usv@bfitvarkappa{"1D752}
104 \def\um@usv@bfitvarphi{"1D753}
105 \def\um@usv@bfitvarrho{"1D754}
106 \def\um@usv@bfitvarpi{"1D755}

```

Nabla:

```

107 \def\um@usv@Nabla{"2207}
108 \def\um@usv@itNabla{"1D6FB}
109 \def\um@usv@bfNabla{"1D6C1}
110 \def\um@usv@bfitNabla{"1D735}
111 \def\um@usv@bfsfNabla{"1D76F}
112 \def\um@usv@bfsfitNabla{"1D7A9}

```

Partial:

```

113 \def\um@usv@partial{"2202}
114 \def\um@usv@itpartial{"1D715}
115 \def\um@usv@bfpartial{"1D6DB}
116 \def\um@usv@bfitpartial{"1D74F}
117 \def\um@usv@bfsfpartial{"1D789}
118 \def\um@usv@bfsfitpartial{"1D7C3}

```

4.1 Package options

xkeyval's package support is used here.

math-style

```

119 \define@choicekey*{unicode-math.sty}
120   {math-style}[\@tempa\@tempb]{iso,tex,french,literal}{
121   \ifcase\@tempb\relax
122     \um@upGreekfalse
123     \um@upgreekfalse
124     \um@upLatinfalse
125     \um@uplatinfalse
126     \um@bfupGreekfalse
127     \um@bfupgreekfalse
128     \um@uppartialfalse
129     \um@bfupLatinfalse
130     \um@bfuplatinfalse
131     \um@upNablafalse
132     \bool_set_false:N \g_um_upsans_bool
133     \bool_set_false:N \g_um_texgreek_bool
134   \or
135     \um@upGreektrue
136     \um@upgreekfalse
137     \um@upLatinfalse
138     \um@uplatinfalse
139     \um@bfupGreektrue
140     \um@bfupgreekfalse
141     \um@uppartialfalse
142     \um@bfupLatintrue
143     \um@bfuplatintrue
144     \um@upNabltrue
145     \bool_set_true:N \g_um_upsans_bool
146     \bool_set_true:N \g_um_texgreek_bool
147   \or
148     \um@upGreektrue
149     \um@upgreektrue
150     \um@upLatintrue
151     \um@uplatinfalse
152     \um@bfupGreektrue

```



```

153 \um@bfupgreektrue
154 \um@uppartialtrue
155 \um@bfupLatintrue
156 \um@bfuplatintrue
157 \um@upNablatrue
158 \bool_set_true:N \g_um_upsans_bool
159 \bool_set_false:N \g_um_texgreek_bool
160 \or
161 \um@literaltrue
162 \um@bfliteraltrue
163 \bool_set_true:N \g_um_sfliteral_bool
164 \bool_set_false:N \g_um_texgreek_bool
165 \fi
166 }

```

bold-style

```

167 \define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,french,literal}{
168 \ifcase\@tempb\relax
169 \um@bfupGreekfalse
170 \um@bfupgreekfalse
171 \um@uppartialfalse
172 \um@bfupLatinfalse
173 \um@bfuplatinfalse
174 \or
175 \um@bfupGreektrue
176 \um@bfupgreekfalse
177 \um@uppartialfalse
178 \um@bfupLatintrue
179 \um@bfuplatintrue
180 \or
181 \um@bfupGreektrue
182 \um@bfupgreektrue
183 \um@uppartialtrue
184 \um@bfupLatintrue
185 \um@bfuplatintrue
186 \or
187 \um@bfliteraltrue
188 \fi
189 }

```

sans-style

```

190 \bool_new:N \g_um_upsans_bool
191 \bool_new:N \g_um_sfliteral_bool
192 \define@choicekey*{unicode-math.sty}
193 {sans-style}[\@tempa\@tempb]{iso,tex,literal}{

```

```

194 \ifcase\@tempb\relax
195   \bool_set_false:N \g_um_upsans_bool
196 \or
197   \bool_set_true:N \g_um_upsans_bool
198 \or
199   \bool_set_true:N \g_um_sfliteral_bool
200 \fi
201 }

```

Symbol obliqueness

```

202 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
203   \ifcase\@tempb\relax
204     \@um@upNabla>true
205   \or
206     \@um@upNabla>false
207   \fi
208 }
209 \cs_set:Nn \um_setup_nabla: {
210   \if@um@upNabla
211     \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@Nabla }
212     \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfNabla }
213     \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfNabla }
214   \else
215     \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@itNabla }
216     \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfitNabla }
217     \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfitNabla }
218   \fi
219 }
220 \define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
221   \ifcase\@tempb\relax
222     \@um@uppartial>true
223   \or
224     \@um@uppartial>false
225   \fi
226 }
227 \cs_set:Nn \um_setup_partial: {
228   \if@um@uppartial
229     \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@partial }
230     \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfpartial }
231     \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfpartial }
232   \else
233     \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@itpartial }
234     \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfitpartial }
235     \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfitpartial }
236   \fi
237 }

```

Epsilon and phi shapes

```
238 \define@choicekey*{unicode-math.sty}{vargreek-shape}[\@tempa\@tempb]{unicode,TeX}{
239   \ifcase\@tempb\relax
240     \bool_set_false:N \g_um_texgreek_bool
241   \or
242     \bool_set_true:N \g_um_texgreek_bool
243   \fi
244 }
```

Colon style

```
245 \bool_new:N \g_um_literal_colon_bool
246 \define@choicekey*{unicode-math.sty}{colon}[\@tempa\@tempb]{literal,TeX}{
247   \ifcase\@tempb\relax
248     \bool_set_true:N \g_um_literal_colon_bool
249   \or
250     \bool_set_false:N \g_um_literal_colon_bool
251   \fi
252 }
253 \ExecuteOptionsX{math-style=TeX}
254 \ProcessOptionsX
```

4.2 Overcoming \@onlypreamble

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

```
255 \tl_map_inline:nn {
256   \new@mathgroup
257   \cdp@list
258   \cdp@elt
259   \DeclareMathSizes
260   \@DeclareMathSizes
261   \newmathalphabet
262   \newmathalphabet@
263   \newmathalphabet@@
264   \DeclareMathVersion
265   \define@mathalphabet
266   \define@mathgroup
267   \addtoversion
268   \version@list
269   \version@elt
270   \alpha@list
271   \alpha@elt
272   \restore@mathversion
273   \init@restore@version
274   \dorestore@version
```

```

275 \process@table
276 \new@mathversion
277 \DeclareSymbolFont
278 \group@list
279 \group@elt
280 \new@symbolfont
281 \SetSymbolFont
282 \SetSymbolFont@
283 \get@cdp
284 \DeclareMathAlphabet
285 \new@mathalphabet
286 \SetMathAlphabet
287 \SetMathAlphabet@
288 \DeclareMathAccent
289 \set@mathaccent
290 \DeclareMathSymbol
291 \set@mathchar
292 \set@mathsymbol
293 \DeclareMathDelimiter
294 \@xxDeclareMathDelimiter
295 \@DeclareMathDelimiter
296 \@xDeclareMathDelimiter
297 \set@mathdelimiter
298 \set@@mathdelimiter
299 \DeclareMathRadical
300 \mathchar@type
301 \DeclareSymbolFontAlphabet
302 \DeclareSymbolFontAlphabet@
303 }{
304 \tl_remove_in:Nn \@preamblecmds {\do#1}
305 }

```

4.3 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
0.60
0.65

```

\font\tmpfont="Cambria Math"
\um@fontdimen@percent{10}{\tmpfont}\
\um@fontdimen@percent{11}{\tmpfont}\
\um@fontdimen@percent{65}{\tmpfont}

```

```

306 \def\um@fontdimen@percent#1#2{
307 0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax

```

```

308 }

\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.
309 \def\um@scaled@apply#1#2#3{
310   \ifx#1\scriptstyle
311     #2\um@fontdimen@percent{10}\um@font#3
312   \else
313     \ifx#1\scriptscriptstyle
314       #2\um@fontdimen@percent{11}\um@font#3
315     \else
316       #2#3%
317     \fi
318   \fi
319 }

```

5 Fundamentals

5.1 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

```

320 \def\new@mathgroup{\alloc@8\mathgroup\chardef\@cclvi}
321 \let\newfam\new@mathgroup

```

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

5.2 `\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
322 \def \um@mathsymbol#1#2#3#4{
323   \expandafter\um@set@mathsymbol\csname sym#3\endcsname#1#2{#4}}

```

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

```

\um@set@mathsymbol #1 : Symbol font number
#2 : Symbol macro, e.g., \alpha
#3 : Type, e.g., \mathalpha
#4 : Slot, e.g., "221E
If the symbol definition is for a macro. There are a bunch of tests to perform to
process the various characters.
324 \def\um@set@mathsymbol#1#2#3#4{

```

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

```

325 \ifx\mathop#3\relax
In order for literal unicode characters to be used in the source and still have the
correct limits behaviour, big operators are made math-active.
The active math char is defined to expand to the macro \sumop.
326 \begingroup
327 \char_make_active:n {#4}
328 \global\mathcode#4="8000\relax
329 \um@scanactivedef #4 \@nil { \csname\cs_to_str:N #2 op\endcsname }
330 \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`.

```

331 \expandafter\global\expandafter\XeTeXmathchardef
332 \csname\string#2@sym\endcsname
333 ="\mathchar@type#3 #1 #4\relax

```

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

```

334 \cs_gset:cpn { \cs_to_str:N #2 op } {
335 \csname\string#2@sym\endcsname
336 \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
337 \ifin@
338 \expandafter\nolimits
339 \fi
340 }

```

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

```

341 \else

```

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

```

342 \expandafter\in@\expandafter#2\expandafter{\um@radicals,}
343 \ifin@

```

```

344     \cs_gset:cpn {\cs_to_str:N #2 sign} { \XeTeXradical #1 #4 \relax }
345     \else

```

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

```

346     \ifx\mathopen#3\relax
347     \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
348     \global\XeTeXdelcode#4=#1 #4\relax
349     \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
350     \else
351     \ifx\mathclose#3\relax
352     \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
353     \global\XeTeXdelcode#4=#1 #4\relax
354     \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
355     \else

```

Accents

```

356     \ifx\mathaccent#3\relax
357     \cs_gset:Npx #2 {\XeTeXmathaccent "\mathchar@type#3 #1 #4\relax}
358     \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.

```

359     \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
360     \fi
361     \fi
362     \fi
363     \fi
364     \fi
365 }

```

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

```

366 \cs_set:Nn \um_set_mathcode:nnnn {
367   \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
368 }

```

5.3 The main `\setmathfont` macro

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

```

369 \DeclareDocumentCommand \setmathfont { O{} m } {

```

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

```
370 \let\glb@currsize\relax
```

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

```
371 \let\um@char@range\@empty
372 \let\um@char@num@range\@empty
```

- Tell fontspec that maths font features are actually allowed.

```
373 \um@fontspec@featuretrue
```

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

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

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

```
375 \def\um@mversion{normal}
376 \DeclareMathVersion{\um@mversion}
```

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

```
377 \tl_set:Nn \l_um_script_features_tl {ScriptStyle}
378 \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
379 \tl_set:Nn \l_um_script_font_tl {#2}
380 \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.

```
381 \setkeys*[um]{options}{#1}
382 \edef\@tempa{\noexpand\zf@fontspec{
383   Script = Math,
384   SizeFeatures = {
385     {Size = \tf@size-} ,
386     {Size = \sf@size-\tf@size ,
387       Font = \l_um_script_font_tl ,
388       \l_um_script_features_tl
389     } ,
390     {Size = -\sf@size ,
391       Font = \l_um_sscript_font_tl ,
```



```

392         \l_um_sscript_features_tl
393     },
394     \XKV@rm
395     }{#2}
396 }
397 \@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`:

```

399 \font\um@font="#2"\relax
400 %% \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
401 %% \um@ot@math@true
402 %% \else
403 %% \PackageWarningNoLine{unicode-math}{
404 %%     The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
405 %%     Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
406 %%     in~ a~ substandard~ manner
407 %% }
408 %% \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 §5.3.1 for the individual definitions

```

409 \ifx\um@char@range\@empty
410     \tl_set:Nn \um_symfont_tl {um@allsym}
411     \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
412     \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
413     \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
414     \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
415     \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
416 \else
417     \stepcounter{um@fam}
418     \tl_set:Nx \um_symfont_tl {um@fam\theum@fam}
419     \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn
420     \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
421     \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
422     \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
423 \fi

```

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

```

424 \DeclareSymbolFont{\um_symfont_tl}
425     {\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`.

```
426 \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.)

```
427 \um_setup_shapes:
```

```
428 \um_remap_symbols:
```

```
429 \um_setup_mathactives:
```

```
430 \um_setup_alphanum:
```

```
431 \um_setup_alphabets:
```

End of the `\setmathfont` macro.

```
432 }
```

```
433 \cs_new:Nn \um_setup_shapes: {
```

```
434 \um_setup_nabla:
```

```
435 \um_setup_partial:
```

```
436 }
```

5.3.1 Functions for setting up symbols with mathcodes

`\um_process_symbol_noparse:nnnn` If the Range font feature has been used, then only a subset of the unicode glyphs are to be defined. See section §6.3 for the code that enables this.

`\um_process_symbol_parse:nnnn`

```
437 \cs_set:Nn \um_process_symbol_noparse:nnnn {
```

```
438 \um@mathsymbol{#2}{#3}{\um_symfont_tl}{#1}
```

```
439 }
```

```
440 \cs_set:Nn \um_process_symbol_parse:nnnn {
```

```
441 \um@parse@term{#1}{#2}{#3}{
```

```
442 \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
```

```
443 }
```

```
444 }
```

`\um_remap_symbols:`

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

`\um_remap_symbol_noparse:nnn`

`\um_remap_symbol_parse:nnn`

```
445 \cs_new:Nn \um_remap_symbols: {
```

```
446 \um_remap_symbol:nnn{'\-}{\mathbin}{"02212}% hyphen to minus
```

```

447 \um_remap_symbol:nnn{\`*}{\mathbin}{"02217}% text asterisk to "centred as-
    terisk"
448 \bool_if:NF \g_um_literal_colon_bool {
449   \um_remap_symbol:nnn{\`:}{\mathrel}{"02236}% colon to ratio (i.e., punct to rel)
450 }
451 \if@um@literal
452   \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
453   \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
454   \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
455   \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
456 \else
457   \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um@Nabla_up_or_it_usv}
458   \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um@partial_up_or_it_usv}
459 \fi

```

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

```

460 \if@um@bfliteral
461   \um_remap_symbol:nnn {\um@usv@bfNabla}{\mathord}{\um@usv@bfNabla}
462   \um_remap_symbol:nnn {\um@usv@bfitNabla}{\mathord}{\um@usv@bfitNabla}
463   \um_remap_symbol:nnn {\um@usv@bfsfNabla}{\mathord}{\um@usv@bfsfNabla}
464   \um_remap_symbol:nnn {\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsfitNabla}
465   \um_remap_symbol:nnn {\um@usv@bfpartial}{\mathord}{\um@usv@bfpartial}
466   \um_remap_symbol:nnn {\um@usv@bfitpartial}{\mathord}{\um@usv@bfitpartial}
467   \um_remap_symbol:nnn {\um@usv@bfsfpartial}{\mathord}{\um@usv@bfsfpartial}
468   \um_remap_symbol:nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
469 \else
470   \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla}{\mathord}{\um@bfNabla_up_or_it_usv}
471   \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla}{\mathord}{\um@bfsfNabla_up_or_it_usv}
472   \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um@bfpartial_up_or_it_usv}
473   \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um@bfsfpartial_up_or_it_usv}
474 \fi
475 }

```

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

```

476 \cs_new:Nn \um_remap_symbol_parse:nnn {
477   \um@parse@term {#3} {\@nil} {#2} {
478     \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
479   }
480 }
481 \cs_new:Nn \um_remap_symbol_noparse:nnn {
482   \clist_map_inline:nn {#1} {
483     \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
484   }
485 }

```

5.3.2 Active math characters

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

`\um_setup_mathactives:`

```
486 \cs_new:Nn \um_setup_mathactives: {  
487   \um_make_mathactive:nNN {"2032} \primesingle \mathord  
488 }
```

`\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!

```
489 \cs_new:Nn \um_make_mathactive:nNN {  
490   \XeTeXmathchardef #2 = "\mathchar@type #3  
491                               \csname sym\um_symfont_tl\endcsname  
492                               #1 \scan_stop:  
493   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:  
494 }
```

5.3.3 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.

```
495 \cs_set:Nn \um_setup_alphanum: {  
496   \ifx\um@char@range\@empty  
497     \um_map_chars_numbers:nn {\um@usv@num}{\um@usv@num}
```

Normal weight

```
498   \ifum@literal  
499     \um_setup_literals:  
500   \else  
501     \um_setup_Latin:  
502     \um_setup_latin:  
503     \um_setup_Greek:  
504     \um_setup_greek:  
505   \fi
```

Bold

```
506   \ifum@bfliteral  
507     \um_setup_bf_literals:  
508   \else  
509     \ifum@bfupLatin
```

```

510 \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfupLatin}
511 \else
512 \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
513 \fi
514 \if@um@bfuplatin
515 \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfuplatin}
516 \else
517 \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
518 \fi
519 \if@um@bfupgreek
520 \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfupGreek}
521 \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
522 \else
523 \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}
524 \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
525 \fi
526 \if@um@bfupgreek
527 \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
528 \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
529 \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
530 \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
531 \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
532 \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
533 \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
534 \else
535 \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
536 \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
537 \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
538 \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
539 \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
540 \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
541 \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
542 \fi
543 \fi
544 \else
: TODO : what is supposed to happen here?
545 \fi
546 }

```

5.3.4 Functions for setting up the maths alphabets

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

of the current symbol font.

```

547 \cs_set:Nn \um_mathmap_noparse:Nnn {
548   \clist_map_inline:nn {#2} {
549     \exp_args:No \um@addto@mathmap \um_symfont_tl {##1}{#1}{#3}
550   }
551 }

```

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

```

552 \cs_set:Nn \um_mathmap_parse:Nnn {
553   \clist_map_inline:Nn \um@char@num@range {
554     \ifnum##1=##3\relax
555       \clist_map_inline:nn {#2} {
556         \exp_args:No \um@addto@mathmap \um_symfont_tl {####1}{#1}{#3}
557       }
558     \fi
559   }
560 }

```

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

```

561 \newcommand\um@addto@mathmap[4]{
562   \expandafter\g@addto@macro
563     \csname um_setup\cs_to_str:N #3:\endcsname{
564     \um_set_mathcode:nnnn{#2}{\mathalpha}{#1}{#4}
565   }
566 }

```





























5.4 (Big) operators









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


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

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

USV	Ex.	Macro	Description
U+02140	$\sum\limits_0^1$	<code>\Bbbsum</code>	DOUBLE-STRUCK N-ARY SUMMATION
U+0220F	$\prod\limits_0^1$	<code>\prod</code>	PRODUCT OPERATOR
U+02210	$\coprod\limits_0^1$	<code>\coprod</code>	COPRODUCT OPERATOR
U+02211	$\sum\limits_0^1$	<code>\sum</code>	SUMMATION OPERATOR
U+0222B	$\int\limits_0^1$	<code>\int</code>	INTEGRAL OPERATOR
U+0222C	$\iint\limits_0^1$	<code>\iint</code>	DOUBLE INTEGRAL OPERATOR
U+0222D	$\iiint\limits_0^1$	<code>\iiint</code>	TRIPLE INTEGRAL OPERATOR
U+0222E	$\oint\limits_0^1$	<code>\oint</code>	CONTOUR INTEGRAL OPERATOR
U+0222F	$\oiint\limits_0^1$	<code>\oiint</code>	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230	$\oiiint\limits_0^1$	<code>\oiiint</code>	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231	$\int\limits_0^1$	<code>\intclockwise</code>	CLOCKWISE INTEGRAL
U+02232	$\oint\limits_0^1$	<code>\varointclockwise</code>	CONTOUR INTEGRAL, CLOCKWISE
U+02233	$\oint\limits_0^1$	<code>\ointctrclockwise</code>	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0	$\bigwedge\limits_0^1$	<code>\bigwedge</code>	LOGICAL OR OPERATOR
U+022C1	$\bigvee\limits_0^1$	<code>\bigvee</code>	LOGICAL AND OPERATOR
U+022C2	$\bigcap\limits_0^1$	<code>\bigcap</code>	INTERSECTION OPERATOR
U+022C3	$\bigcup\limits_0^1$	<code>\bigcup</code>	UNION OPERATOR
U+027D5	$\leftthreetimes\limits_0^1$	<code>\leftouterjoin</code>	LEFT OUTER JOIN
U+027D6	$\rightthreetimes\limits_0^1$	<code>\rightouterjoin</code>	RIGHT OUTER JOIN
U+027D7	$\frown\limits_0^1$	<code>\fullouterjoin</code>	FULL OUTER JOIN
U+027D8	$\bot\limits_0^1$	<code>\bigbot</code>	LARGE UP TACK
U+027D9	$\top\limits_0^1$	<code>\bigtop</code>	LARGE DOWN TACK
U+029F8	$\big/$	<code>\xsol</code>	BIG SOLIDUS

U+029F9		\xbsol	BIG REVERSE SOLIDUS
U+02A00		\bigodot	N-ARY CIRCLED DOT OPERATOR
U+02A01		\bigoplus	N-ARY CIRCLED PLUS OPERATOR
U+02A02		\bigotimes	N-ARY CIRCLED TIMES OPERATOR
U+02A03		\bigcupdot	N-ARY UNION OPERATOR WITH DOT
U+02A04		\biguplus	N-ARY UNION OPERATOR WITH PLUS
U+02A05		\bigsqcap	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06		\bigsqcup	N-ARY SQUARE UNION OPERATOR
U+02A07		\conjquant	TWO LOGICAL AND OPERATOR
U+02A08		\disjquant	TWO LOGICAL OR OPERATOR
U+02A09		\bigtimes	N-ARY TIMES OPERATOR
U+02A0B		\sumint	SUMMATION WITH INTEGRAL
U+02A0C		\iiint	QUADRUPLE INTEGRAL OPERATOR
U+02A0D		\intbar	FINITE PART INTEGRAL
U+02A0E		\intBar	INTEGRAL WITH DOUBLE STROKE
U+02A0F		\fint	INTEGRAL AVERAGE WITH SLASH
U+02A10		\cirfnint	CIRCULATION FUNCTION
U+02A11		\awint	ANTICLOCKWISE INTEGRATION LINE INTEGRATION WITH RECTANGULAR
U+02A12		\rppointint	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13		\scpointint	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE
U+02A14		\npointint	POLE
U+02A15		\pointint	INTEGRAL AROUND A POINT OPERATOR
U+02A16		\sqint	QUATERNION INTEGRAL OPERATOR INTEGRAL WITH LEFTWARDS ARROW WITH
U+02A17		\intlarhk	HOOK
U+02A18		\intx	INTEGRAL WITH TIMES SIGN
U+02A19		\intcap	INTEGRAL WITH INTERSECTION
U+02A1A		\intcup	INTEGRAL WITH UNION
U+02A1B		\intoverbar	INTEGRAL WITH OVERBAR

U+02A1C		<code>\lowint</code>	INTEGRAL WITH UNDERBAR
U+02A1D		<code>\Join</code>	JOIN
U+02A1E		<code>\bigtriangleleft</code>	LARGE LEFT TRIANGLE OPERATOR
U+02A1F		<code>\zcmp</code>	Z NOTATION SCHEMA COMPOSITION
U+02A20		<code>\zpipe</code>	Z NOTATION SCHEMA PIPING
U+02A21		<code>\zproject</code>	Z NOTATION SCHEMA PROJECTION
U+02AFC		<code>\biginterleave</code>	LARGE TRIPLE VERTICAL BAR OPERATOR
U+02AFF		<code>\bigtalloblong</code>	N-ARY WHITE VERTICAL BAR

`\um@nolimits` This macro is a sequence containing those maths operators that require a `\nolimits` suffix. This list is used when processing `unicode-math-table.tex` to define such commands automatically (see the macro `\um@set@mathsymbol` on page 22). I’ve chosen essentially just the operators that look like integrals; hopefully a better mathematician can help me out here. I’ve a feeling that it’s more useful *not* to include the multiple integrals such as , but that might be a matter of preference.

```

567 \def\um@nolimits{
568   \@elt\int\@elt\iint\@elt\iiint\@elt\iiint\@elt\oint\@elt\oiint\@elt\oiint
569   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrackwise\@elt\sumint
570   \@elt\intbar\@elt\intBar\@elt\oint\@elt\cirfnint\@elt\awint\@elt\rppoint
571   \@elt\scpolint\@elt\ntpolint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
572   \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
573 }

```

`\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`.

```

574 \newcommand\addnolimits[1]{
575   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
576 }

```

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

```

577 \def\removenolimits#1{
578   \begingroup
579   \def\@elt##1{
580     \ifx##1#1\else
581       \noexpand\@elt\noexpand##1
582     \fi}
583   \xdef\um@nolimits{\um@nolimits}

```

```

584 \endgroup
585 }

```

5.5 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.

```

586 \def\um@radicals{\sqrt}

```

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

```

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

```

5.6 Delimiters

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

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

```

587 \let\left@primitive\left
588 \def\left{\mathopen{}\left@primitive}

```

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

Some symbols that aren't `mathopen`/`mathclose` still need to have delimiter codes assigned.

Set up `delcodes` so that slashes and things can grow if the font supports it. This is just inserted here so the documentation works. It will be generalised soon.

```

589 \XeTeXdelcode"002F =4 "002F % ord
590 \XeTeXdelcode"005C =4 "005C % ord
591 \XeTeXdelcode"2044 =4 "2044 % bin
592 \XeTeXdelcode"2215 =4 "2215 % bin
593 \XeTeXdelcode"2216 =4 "2216 % bin
594 \XeTeXdelcode"29F5 =4 "29F5 % bin

```

Here are all `\mathopen` characters:

USV	Ex.	Macro	Description
U+00028	(<code>\lparen</code>	LEFT PARENTHESIS

U+0005B	[\lbrack	LEFT SQUARE BRACKET
U+0007B	{	\lbrace	LEFT CURLY BRACKET
U+0007C		\lvert	VERTICAL BAR
U+02016		\lVert	DOUBLE VERTICAL BAR
U+0221A	√	\sqrt	RADICAL
U+0221B	³√	\cuberoot	CUBE ROOT
U+0221C	⁴√	\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	⌢	\langledot	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+0007C		\rvert	VERTICAL BAR
U+0007D	}	\rbrace	RIGHT CURLY BRACKET
U+02016		\rVert	DOUBLE VERTICAL BAR
U+02309]̂	\rceil	RIGHT CEILING
U+0230B]̇	\rfloor	RIGHT FLOOR
U+0231D]̋	\urcorner	UPPER RIGHT CORNER
U+0231F]̌	\lrcorner	LOWER RIGHT CORNER
			LIGHT RIGHT TORTOISE SHELL BRACKET
U+02773		\rbrbrak	ORNAMENT
U+027C6	Œ	\rbag	RIGHT S-SHAPED BAG DELIMITER
			MATHEMATICAL RIGHT WHITE SQUARE
U+027E7]]	\rBrack	BRACKET
U+027E9	>	\rangle	MATHEMATICAL RIGHT ANGLE BRACKET
			MATHEMATICAL RIGHT DOUBLE ANGLE
U+027EB	>>	\rAngle	BRACKET
			MATHEMATICAL RIGHT WHITE TORTOISE
U+027ED		\Rbrbrak	SHELL BRACKET
U+02984	}]	\rBrace	RIGHT WHITE CURLY BRACKET
U+02986)	\rParen	RIGHT WHITE PARENTHESIS
U+02988)̂	\rrparenthesis	Z NOTATION RIGHT IMAGE BRACKET
U+0298A)̇	\rrangle	Z NOTATION RIGHT BINDING BRACKET
U+0298C]̂	\rbrackubar	RIGHT SQUARE BRACKET WITH UNDERBAR
			RIGHT SQUARE BRACKET WITH TICK IN
U+0298E]̋	\rbracklrtick	BOTTOM CORNER
			RIGHT SQUARE BRACKET WITH TICK IN TOP
U+02990]̌	\rbrackurtick	CORNER
U+02992	>̂	\rangledot	RIGHT ANGLE BRACKET WITH DOT
U+02994	>̇	\rpangtr	RIGHT ARC GREATER-THAN BRACKET
U+02998)̋	\rblkrbrak	RIGHT BLACK TORTOISE SHELL BRACKET
U+029D9	⋈	\rvzigzag	RIGHT WIGGLY FENCE
U+029DB	⋈̂	\Rvzigzag	RIGHT DOUBLE WIGGLY FENCE
U+029FD	>̌	\rcurvyangle	RIGHT POINTING CURVED ANGLE BRACKET
U+03015		\rbrbrak	RIGHT BROKEN BRACKET
U+03019		\Rbrbrak	RIGHT WHITE TORTOISE SHELL BRACKET

5.7 Maths accents

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

USV	Ex.	Macro	Description
U+00300	̂	\grave	GRAVE ACCENT
U+00301	́	\acute	ACUTE ACCENT
U+00302	̂	\hat	CIRCUMFLEX ACCENT

U+00303	\tilde{x}	<code>\tilde</code>	TILDE
U+00304	\bar{x}	<code>\bar</code>	MACRON
U+00305	\overline{x}	<code>\overbar</code>	OVERBAR EMBELLISHMENT
U+00306	\breve{x}	<code>\breve</code>	BREVE
U+00307	\dot{x}	<code>\dot</code>	DOT ABOVE
U+00308	\ddot{x}	<code>\ddot</code>	DIERESIS
U+00309	$\overset{\circ}{x}$	<code>\ovhook</code>	COMBINING HOOK ABOVE
U+0030A	\circ	<code>\ocirc</code>	RING
U+0030C	\check{x}	<code>\check</code>	CARON
U+00310	candra	<code>\candra</code>	CANDRABINDU (NON-SPACING)
U+00312	turned comma	<code>\oturnedcomma</code>	COMBINING TURNED COMMA ABOVE GREEK PSILI (SMOOTH BREATHING)
U+00313	smooth	<code>\osmooth</code>	(NON-SPACING) GREEK DASIA (ROUGH BREATHING)
U+00314	rough	<code>\orough</code>	(NON-SPACING)
U+00315	commatopright	<code>\ocommatopright</code>	COMBINING COMMA ABOVE RIGHT
U+0031A	droang	<code>\droang</code>	LEFT ANGLE ABOVE (NON-SPACING) COMBINING LONG SOLIDUS
U+00338	not	<code>\not</code>	OVERLAY
U+020D0	\lefttharpoonup	<code>\leftharpoonupaccent</code>	COMBINING LEFT HARPOON ABOVE
U+020D1	\rightarrow	<code>\rightharpoonupaccent</code>	COMBINING RIGHT HARPOON ABOVE
U+020D2	vertical line	<code>\vertoverlay</code>	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6	$\overleftarrow{}$	<code>\overleftarrow</code>	COMBINING LEFT ARROW ABOVE
U+020D7	$\overrightarrow{}$	<code>\overrightarrow</code>	COMBINING RIGHT ARROW ABOVE
U+020DB	three dots	<code>\dddots</code>	COMBINING THREE DOTS ABOVE
U+020DC	four dots	<code>\ddddots</code>	COMBINING FOUR DOTS ABOVE
U+020E1	$\overleftrightarrow{}$	<code>\overleftrightharpoonup</code>	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7	annuity	<code>\annuity</code>	COMBINING ANNUITY SYMBOL
U+020E8	three underdot	<code>\threeunderdot</code>	COMBINING TRIPLE UNDERDOT
U+020E9	wide bridge above	<code>\widebridgeabove</code>	COMBINING WIDE BRIDGE ABOVE COMBINING RIGHTWARDS HARPOON WITH
U+020EC	$\text{underrightharpoonowdown}$	<code>\underrightharpoonowdown</code>	BARB DOWNWARDS COMBINING LEFTWARDS HARPOON WITH
U+020ED	$\text{underleftharpoonowdown}$	<code>\underleftharpoonowdown</code>	BARB DOWNWARDS
U+020EE	underleftarrow	<code>\underleftarrow</code>	COMBINING LEFT ARROW BELOW
U+020EF	underrightarrow	<code>\underrightarrow</code>	COMBINING RIGHT ARROW BELOW
U+020F0	asteraccent	<code>\asteraccent</code>	COMBINING ASTERISK ABOVE

6 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 com-

mands.

```
595 \newcommand\um@zf@feature[2]{
596   \define@key[zf]{options}{#1}[]{
597     \ifum@fontspec@feature
598       #2
599     \else
600       \PackageError{fontspec/unicode-math}
601         {The ‘#1’ font feature can only be used for maths fonts}
602         {The feature you tried to use can only be in commands
603          like \protect\setmathfont}
604     \fi
605   }
606 }
```

6.1 OpenType maths font features

```
607 \um@zf@feature{ScriptStyle}{
608   \zf@update@ff{+ssty=0}
609 }
610 \um@zf@feature{ScriptScriptStyle}{
611   \zf@update@ff{+ssty=1}
612 }
```

6.2 Script and scriptscript font options

```
613 \define@cmdkey[um]{options}[um@]{ScriptFeatures}{}
614 \define@cmdkey[um]{options}[um@]{ScriptScriptFeatures}{}
615 \define@cmdkey[um]{options}[um@]{ScriptFont}{}
616 \define@cmdkey[um]{options}[um@]{ScriptScriptFont}{}

```

6.3 Range processing

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

```
617 \define@choicekey+[um]{options}{Range}[\@tempa\@tempb]{ALL}{
618   \ifcase\@tempb\relax
619     \global\let\um@char@range\@empty
620   \fi
621 }{
622   \xdef\um@char@range{#1}
623 }
```

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 11.

Table 11: 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:

```

624 \newcommand\um@parse@term[4]{
625   \clist_map_variable:NNn \um@char@range \@ii {
626     \unless\ifx\@ii\@empty
627       \@tempswafalse

```

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

```

628     \expandafter\um@firstchar\expandafter{\@ii}
629     \ifx\@tempa\um@backslash
630       \expandafter\ifx\@ii#2\relax
631         \@tempwattrue
632       \else
633         \expandafter\ifx\@ii#3\relax
634           \@tempwattrue
635         \fi
636       \fi

```

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

```

637     \else
638       \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
639     \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.

```

640     \if@tempswa
641       \ifx\um@char@num@range\@empty
642         \g@addto@macro\um@char@num@range{#1}
643       \else
644         \g@addto@macro\um@char@num@range{,#1}
645       \fi

```

```

646         #4%
647         \fi
648     \fi
649 }
650 }
651 \def\um@firstof#1#2\@nil{#1}
652 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
653 \def\um@firstchar#1{\edef\@tempa{\expandafter\um@firstof\string#1\@nil}}

```

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

\um@parse@range Weird syntax. As shown previously in table 11, this macro can be passed four different input types via \um@parse@term.

```

654 \def\um@parse@range#1-#2-#3\@nil#4\@nil{
655     \def\@tempa{#1}
656     \def\@tempb{#2}

```

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

```

657     \expandafter\ifx\expandafter\@marker\@tempb\relax
658         \ifnum#4=#1\relax
659             \@tempswatruetrue
660         \fi
661     \else

```

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

```

662     \ifx\@empty\@tempb
663         \ifnum#4>\numexpr#1-1\relax
664             \@tempswatruetrue
665         \fi
666     \else

```

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

```

667         \ifx\@empty\@tempa
668         \ifnum#4<\numexpr#2+1\relax
669         \@tempwattrue
670         \fi

```

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

```

671     \else
672     \ifnum#4>\numexpr#1-1\relax
673     \ifnum#4<\numexpr#2+1\relax
674     \@tempwattrue
675     \fi
676     \fi
677     \fi
678     \fi
679     \fi
680 }

```

\um_map_char:nn #1 : Number of iterations
 #2 : Starting input char(s)
 #3 : Starting output char
 Loops through character ranges setting \mathcode.

```

681 \cs_set:Nn \um_map_chars_range:nnn {
682   \clist_map_variable:nnN {#2} \l_um_input_num {
683     \prg_stepwise_variable:nnnN{0}{1}{#1} \l_um_incr_num {
684       \um_set_mathcode:nnnn
685       {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
686       {\mathalpha}{\um_symfont_tl}
687       {\numexpr \l_um_incr_num + #3 \relax}
688     }
689   }
690 }
691 \cs_set:Nn \um_map_chars_latin:nn {
692   \um_map_chars_range:nnn {25}{#1}{#2}
693 }
694 \cs_set:Nn \um_map_chars_greek:nn {
695   \um_map_chars_range:nnn {24}{#1}{#2}
696 }
697 \cs_set:Nn \um_map_chars_numbers:nn {
698   \um_map_chars_range:nnn {9}{#1}{#2}
699 }
700 \cs_set:Nn \um_map_char:nn {
701   \um_map_chars_range:nnn {0}{#1}{#2}
702 }

```

```
\um_set_mathalphabet_char:Nnnn #1 : Maths alphabet
                                #2 : Input char(s)
                                #3 : Output char
                                Loops through character ranges setting \mathcode.
703 \cs_set:Npn \exp_args:Nnff {\::n\::f\::f\:::}
704 \cs_new:Nn \um_set_mathalphabet_char:Nnn {
705   \clist_map_variable:nNn {#2} \l_um_input_num {
706     \exp_args:Nnff \um_mathmap:Nnn {#1}
707       {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
708   }
709 }

\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.
710 \cs_new:Nn \um_set_mathalph_range:nNnn {
711   \clist_map_variable:nNn {#3} \l_um_input_num {
712     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
713       \exp_args:Nnff \um_mathmap:Nnn {#2}
714         {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
715         {\number\numexpr \l_um_inc_num + #4 \relax}
716     }
717   }
718 }
719 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
720   \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
721 }
722 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
723   \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
724 }
725 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
726   \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
727 }

BCDBCDEABCDEF

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

6.4 Resolving Greek symbol name control sequences

\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.

```

728 \AtBeginDocument{\um@resolve@greek}
729 \newcommand\um@resolve@greek{
730   \def\Alpha{\mitAlpha}
731   \def\Beta{\mitBeta}
732   \def\Gamma{\mitGamma}
733   \def\Delta{\mitDelta}
734   \def\Epsilon{\mitEpsilon}
735   \def\Zeta{\mitZeta}
736   \def\Eta{\mitEta}
737   \def\Theta{\mitTheta}
738   \def\Iota{\mitIota}
739   \def\Kappa{\mitKappa}
740   \def\Lambda{\mitLambda}
741   \def\Mu{\mitMu}
742   \def\Nu{\mitNu}
743   \def\Xi{\mitXi}
744   \def\Omicron{\mitOmicron}
745   \def\Pi{\mitPi}
746   \def\Rho{\mitRho}
747   \def\varTheta{\mitvarTheta}
748   \def\Sigma{\mitSigma}
749   \def\Tau{\mitTau}
750   \def\Upsilon{\mitUpsilon}
751   \def\Phi{\mitPhi}
752   \def\Chi{\mitChi}
753   \def\Psi{\mitPsi}
754   \def\Omega{\mitOmega}

```

Lowercase:

```

755   \def\alpha{\mitalpha}
756   \def\beta{\mitbeta}
757   \def\gamma{\mitgamma}
758   \def\delta{\mitdelta}
759   \def\epsilon{
760     \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
761   }
762   \def\zeta{\mitzeta}
763   \def\eta{\miteta}
764   \def\theta{\mittheta}
765   \def\iota{\mitiota}
766   \def\kappa{\mitkappa}
767   \def\lambda{\mitlambda}
768   \def\mu{\mitmu}
769   \def\nu{\mitnu}
770   \def\xi{\mitxi}

```

```

771 \def\omicron{\mitomicron}
772 \def\pi{\mitpi}
773 \def\rho{\mitrho}
774 \def\varsigma{\mitvarsigma}
775 \def\sigma{\mitsigma}
776 \def\tau{\mittau}
777 \def\upsilon{\mitupsilon}
778 \def\phi{
779   \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}
780 }
781 \def\chi{\mitchi}
782 \def\psi{\mitpsi}
783 \def\omega{\mitomega}
784 \def\varepsilon{
785   \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
786 }
787 \def\vartheta{\mitvartheta}
788 \def\varkappa{\mitvarkappa}
789 \def\varphi{
790   \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
791 }
792 \def\varrho{\mitvarrho}
793 \def\varpi{\mitvarpi}
794 }

```

6.5 Setting up the mappings

`\um_setup_literals:` : TODO : other literal symbols

```

795 \cs_set:Nn \um_setup_literals: {
796   \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}
797   \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}
798   \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}
799   \um_map_char:nn {\um@usv@ith}{\um@usv@ith}
800   \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}
801   \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}
802   \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}
803   \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}
804   \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}
805 }

```

`\um_setup_bf_literals:` TODO: other literal symbols

```

806 \cs_set:Nn \um_setup_bf_literals: {
807   \um_map_chars_latin:nn {\um@usv@bfupLatin}{\um@usv@bfupLatin}
808   \um_map_chars_latin:nn {\um@usv@bfuplatin}{\um@usv@bfuplatin}
809   \um_map_chars_latin:nn {\um@usv@bfitLatin}{\um@usv@bfitLatin}
810   \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}

```

```

811 \um_map_chars_greek:nn {\um@usv@bfupGreek}{\um@usv@bfupGreek}
812 \um_map_chars_greek:nn {\um@usv@bfupgreek}{\um@usv@bfupgreek}
813 \um_map_chars_greek:nn {\um@usv@bfitGreek}{\um@usv@bfitGreek}
814 \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
815 }

```

\um_setup_Latin:

```

816 \cs_set:Nn \um_setup_Latin: {
817   \if@um@upLatin
818     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
819   \else
820     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
821   \fi
822 }

```

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

```

823 \cs_set:Nn \um_setup_latin: {
824   \if@um@upLatin
825     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
826     \um_map_char:nn {\um@usv@ith}{`\h}
827   \else
828     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
829     \um_map_char:nn {\`\h,\um@usv@ith}{\um@usv@ith}
830   \fi
831 }

```

\um_setup_Greek:

```

832 \cs_set:Nn \um_setup_Greek: {
833   \if@um@upGreek
834     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
835     \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
836   \else
837     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
838     \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}
839   \fi
840 }

```

\um_setup_greek:

```

841 \cs_set:Nn \um_setup_greek: {
842   \if@um@upgreek
843     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
844     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
845     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
846     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
847     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}

```

```

848 \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
849 \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
850 \else
851 \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
852 \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
853 \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
854 \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
855 \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
856 \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
857 \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
858 \fi
859 }

```

7 Maths alphabets mapping definitions

Algorithm for setting alphabet fonts. By default, when Range is empty, we are in *implicit* mode. If Range contains the name of the math alphabet, we are in *explicit* mode and do things slightly differently.

Implicit mode:

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

Explicit mode:

- Only set the alphabets specified.
- Check for the first glyph of the uppercase Latin alphabet to detect if the font contains the alphabet shape in the unicode math plane.
- For unicode math alphabets, overwrite whatever's already there.
- Otherwise, use the ASCII letters instead.

```

860 \cs_new:Nn \um_setup_alphabets: {
861   \um_setup_math_alphabet:nn {up    }{\latin,Latin,greek,Greek}
862   \um_setup_math_alphabet:n {it    }
863   \um_setup_math_alphabet:n {bb    }
864   \um_setup_math_alphabet:nn {scr   }{\latin,Latin}
865   \um_setup_math_alphabet:nn {frak  }{\latin,Latin}

```

```

866 \um_setup_math_alphabet:n {sf    }
867 \um_setup_math_alphabet:n {sfup  }
868 \um_setup_math_alphabet:n {sfit  }
869 \um_setup_math_alphabet:n {tt    }
870 \um_setup_math_alphabet:n {bf    }
871 \um_setup_math_alphabet:n {bfup  }
872 \um_setup_math_alphabet:n {bfit  }
873 \um_setup_math_alphabet:n {bfscr }
874 \um_setup_math_alphabet:n {bffrak}
875 \um_setup_math_alphabet:n {bfsf  }
876 \um_setup_math_alphabet:n {bfsfup}
877 \um_setup_math_alphabet:n {bfsfit}
878 }

```

`\um_setup_math_alphabet:nn` #1 : Math font family name (e.g., ‘sf’)
 #2 : Math alphabets, comma separated of {latin,Latin,greek,Greek,num}
 First check that at least one of the alphabets for the font shape is defined, and then
 then loop through them defining the individual ranges.

```

879 \cs_new:Nn \um_setup_math_alphabet:nn {
880   \clist_map_inline:nn {#2} {
881     \um_glyph_if_exist:nT {\csname um@usv@#1##1 \endcsname}{
882       \um_maybe_init_alphabet:n {#1}
883       \um_prepare_alph:n {#1}
884       \clist_map_break:
885     }
886   }
887   \clist_map_inline:nn {#2} {
888     \um_glyph_if_exist:nTF {\csname um@usv@#1##1 \endcsname}{
889       \use:c {um_config_math#1_#1:}
890     }{
891       \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
892         Math~ alphabet~
893         \@backslashchar math#1~
894         (\tl_use:c{g_um_math_alphabet_name_#1_tl})~
895         not~ found~ in~ font~
896         \fontname\um@font}
897     }
898   }
899 }
900 \tl_set:Nn \g_um_math_alphabet_name_latin_tl {Latin, lowercase}
901 \tl_set:Nn \g_um_math_alphabet_name_Latin_tl {Latin, uppercase}
902 \tl_set:Nn \g_um_math_alphabet_name_greek_tl {Greek, lowercase}
903 \tl_set:Nn \g_um_math_alphabet_name_Greek_tl {Greek, uppercase}
904 \tl_set:Nn \g_um_math_alphabet_name_num_tl {Numerals}

905 \cs_new:Nn \um_setup_math_alphabet:n {
906   \um_glyph_if_exist:nTF {\csname um@usv@#1Latin \endcsname}{

```

```

907 \um_maybe_init_alphabet:n {#1}
908 \um_prepare_alph:n {#1}
909 \use:c {um_config_math#1:}
910 }{
911 \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
912 Math~ alphabet~ \@backslashchar math#1~ not~ found~ in~ font~ \font-
name\um@font}
913 \cs_if_exist:cT {um_fix_math#1:} {
914 \use:c {um_fix_math#1:}
915 }
916 }
917 }
918 \cs_set:Nn \um_fix_mathtt: {
919 \SetMathAlphabet\mathtt{normal}\encodingdefault\ttdefault\mddefault\updefault
920 }
921 \cs_set:Nn \um_init_alphabet:n {
922 \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
923 }

```

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

```

924 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
925 \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
926 }

```

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

```

927 \cs_new:Nn \um_prepare_alph:n {
928 \cs_if_exist:cF {um_math#1:n} {
929 \cs_set:cpn {um_math#1:n} ##1 {
930 \use:c {um_setup_math#1:} ##1 \egroup
931 }
932 \cs_set_protected:cpn {math#1} {
933 \bgroup
934 \mode_if_math:F {
935 \egroup\expandafter
936 \non@alpherr\expandafter{\csname math#1\endcsname\space}
937 }
938 \use:c {um_math#1:n}
939 }
940 }
941 }

```

: TODO : nested alphabets?

7.1 Non-bold math alphabets

7.1.1 Upright: `\mathup`

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

```
942 \cs_new:Npn \um_config_mathup_Latin: {  
943   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}  
944 }  
945 \cs_new:Npn \um_config_mathup_latin: {  
946   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}  
947 }  
948 \cs_new:Npn \um_config_mathup_Greek: {  
949   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}  
950   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}  
951   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varTheta}  
952 }  
953 \cs_new:Npn \um_config_mathup_greek: {  
954   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}  
955   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}  
956   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}  
957   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}  
958   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}  
959   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}  
960   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}  
961   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}  
962 }
```

7.1.2 Italic: `\mathit`

Roman:

```
963 \cs_new:Npn \um_config_mathit: {  
964   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}  
965   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}  
966   \um_set_mathalphabet_char:Nnn{\mathit}{\h,\um@usv@ith}{\um@usv@ith}
```

Greek:

```
967   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}  
968   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}  
969   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}  
970   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartial}  
971   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarTheta}  
972   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}  
973   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}  
974   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}  
975   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}  
976   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}  
977   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
```

978 }

7.1.3 Blackboard or double-struck: `\mathbb`

```
979 \cs_new:Npn \um_config_mathbb: {  
980   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@num}{\um@usv@bbnum}  
981   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}  
982   \um_set_mathalphabet_char:Nnn{\mathbb}{`\C,"1D60A}{"2102}  
983   \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D60F}{"210D}  
984   \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D60F}{"2115}  
985   \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D617}{"2119}  
986   \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D618}{"211A}  
987   \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D619}{"211D}  
988   \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D621}{ "2124}  
989   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}  
990 }
```

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

```
991 \cs_new:Npn \um_config_mathscr_Latin: {  
992   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}  
993   \um_set_mathalphabet_char:Nnn \mathscr {\`\B,"1D435}{ "212C}  
994   \um_set_mathalphabet_char:Nnn \mathscr {\`\E,"1D438}{ "2130}  
995   \um_set_mathalphabet_char:Nnn \mathscr {\`\F,"1D439}{ "2131}  
996   \um_set_mathalphabet_char:Nnn \mathscr {\`\H,"1D43B}{ "210B}  
997   \um_set_mathalphabet_char:Nnn \mathscr {\`\I,"1D43C}{ "2110}  
998   \um_set_mathalphabet_char:Nnn \mathscr {\`\L,"1D43F}{ "2112}  
999   \um_set_mathalphabet_char:Nnn \mathscr {\`\M,"1D440}{ "2133}  
1000   \um_set_mathalphabet_char:Nnn \mathscr {\`\R,"1D445}{ "211B}  
1001 }  
1002 \cs_new:Npn \um_config_mathscr_latin: {  
1003   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrlatin}  
1004   \um_set_mathalphabet_char:Nnn \mathscr {\`\e,"1D452}{ "212F}  
1005   \um_set_mathalphabet_char:Nnn \mathscr {\`\g,"1D454}{ "210A}  
1006   \um_set_mathalphabet_char:Nnn \mathscr {\`\o,"1D45C}{ "2134}  
1007 }
```

7.1.5 Fraktur or fraktur or blackletter: `\mathfrak`

```
1008 \cs_new:Npn \um_config_mathfrak_Latin: {  
1009   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLatin}  
1010   \um_set_mathalphabet_char:Nnn \mathfrak {\`\C,"1D436}{ "212D}  
1011   \um_set_mathalphabet_char:Nnn \mathfrak {\`\H,"1D43B}{ "210C}  
1012   \um_set_mathalphabet_char:Nnn \mathfrak {\`\I,"1D43C}{ "2111}  
1013   \um_set_mathalphabet_char:Nnn \mathfrak {\`\R,"1D445}{ "211C}  
1014   \um_set_mathalphabet_char:Nnn \mathfrak {\`\Z,"1D44D}{ "2128}  
1015 }  
1016 \cs_new:Npn \um_config_mathfrak_latin: {
```

```

1017 \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklatin}
1018 }

```

7.1.6 Sans serif: \mathsf

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

```

1019 \cs_new:Npn \um_config_mathsf: {
1020   \bool_if:NTF \g_um_sfliteral_bool {
1021     \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@enum}{\um@usv@sfnum}
1022     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin}{\um@usv@sfupLatin}
1023     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin}{\um@usv@sfuplatin}
1024     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itLatin}{\um@usv@sfitLatin}
1025     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itlatin}{\um@usv@sfitlatin}
1026   }{
1027     \bool_if:NTF \g_um_upsans_bool {
1028       \um_set_mathalphabet_numbers:Nnn \mathsf {\um@usv@enum}{\um@usv@sfnum}
1029       \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfupLatin}
1030       \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfuplatin}
1031     }{
1032       \um_set_mathalphabet_numbers:Nnn \mathsf {\um@usv@enum}{\um@usv@sfnum}
1033       \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLatin}
1034       \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlatin}
1035     }
1036   }
1037 }

```

7.1.7 Sans serif upright: \mathsfup

```

1038 \cs_new:Npn \um_config_mathsfup: {
1039   \um_set_mathalphabet_numbers:Nnn{\mathsfup}{\um@usv@enum}{\um@usv@sfnum}
1040   \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfupLatin}
1041   \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfuplatin}
1042 }

```

7.1.8 Sans serif italic: \mathsfit

Map the numbers like that because it seems sensible.

```

1043 \cs_new:Npn \um_config_mathsfrit: {
1044   \um_set_mathalphabet_numbers:Nnn{\mathsfit}{\um@usv@enum}{\um@usv@sfnum}
1045   \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLatin}
1046   \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlatin}
1047 }

```

7.1.9 Typewriter or monospaced: `\mathtt`

```

1048 \cs_new:Npn \um_config_mathtt: {
1049   \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
1050   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
1051   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlatin}
1052 }

```

7.2 Bold math alphabets

7.2.1 Bold: `\mathbf`

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
Θ
αβγδεζηθικλμνξοπρστυφχψω
εθκφρω

```

 $\mathbf{0123456789}$  $ \
 $\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$  $ \
 $\mathbf{abcdefghijklmnopqrstuvwxyz}$  $ \
 $\mathbf{\Theta}$  $ \quad  $\mathbf{\Theta}$  $ \
 $\mathbf{\alpha\beta\gamma\delta\epsilon\zeta\eta\theta\iota\kappa\lambda\mu\nu\xi\omicron\pi\rho\sigma\tau\upsilon\phi\chi\psi\omega}$  $ \quad  $\mathbf{\epsilon\theta\kappa\phi\rho\omega}$  $ \

```

```

1053 \cs_new:Npn \um_config_mathbf: {
1054   \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
1055   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
1056   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
1057   \if@um@bfliteral
1058     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfupLatin}
1059     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
1060     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@bfuplatin}
1061     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@bfitlatin}
1062     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfupGreek}
1063     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
1064     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfupgreek}
1065     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitgreek}
1066     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1067     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
1068     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
1069     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
1070     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
1071     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bfvarepsilon}
1072     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
1073     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
1074     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
1075     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
1076     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}
1077     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
1078     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}

```

```

1079 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
1080 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
1081 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
1082 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
1083 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1084 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
1085 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
1086 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
1087 \else
1088   \if@um@bfupLatin
1089     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLatin}
1090   \else
1091     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1092   \fi
1093   \if@um@bfuplatin
1094     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplatin}
1095     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
1096   \else
1097     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
1098     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1099   \fi
1100   \if@um@bfupGreek
1101     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGreek}
1102     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarT}
1103   \else
1104     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
1105     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitva}
1106   \fi
1107   \if@um@bfupgreek
1108     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgreek}
1109     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf}
1110     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvarT}
1111     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvark}
1112     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1113     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1114     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1115     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartia}
1116   \else
1117     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}
1118     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf}
1119     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitva}
1120     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitva}
1121     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi}
1122     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho}
1123     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1124     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpart}

```

```

1125 \fi
1126 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um@bfNabla_up_or_it_
1127 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@bfpartial_up_
1128 \fi
1129 }

```

7.2.2 Bold Italic: \mathbfit

0123456789	<code>\$\mathbfit{0123456789}\$ \\\</code>
ABCDEFGHIJKLMNOPQRSTUVWXYZ	<code>\$\mathbfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\\</code>
abcdefghijklmnopqrstuvwxyz	<code>\$\mathbfit{abcdefghijklmnopqrstuvwxyz}\$ \\\</code>
ΑΒΓΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	<code>\$\mathbfit{ }\$ \\\</code>
αβγδεζηθικλμνξοπρστυφχψω εθκφρπ	<code>\$\mathbfit{ }\$ \\\</code>

```

1130 \cs_new:Npn \um_config_mathbfit: {
1131   \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@enum}{\um@usv@bfnun}
1132   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1133   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
1134   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
1135   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}
1136   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfupLatin}{\um@usv@bfitLatin}
1137   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfuplatin}{\um@usv@bfitlatin}
1138   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfupGreek}{\um@usv@bfitGreek}
1139   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfupgreek}{\um@usv@bfitgreek}
1140   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitvar}
1141   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}
1142   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpart}
1143   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf}
1144   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitvar}
1145   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitvar}
1146   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi}
1147   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho}
1148   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1149 }

```

7.2.3 Bold Italic: \mathbfup

0123456789	<code>\$\mathbfup{0123456789}\$ \\\</code>
ABCDEFGHIJKLMNOPQRSTUVWXYZ	<code>\$\mathbfup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}\$ \\\</code>
abcdefghijklmnopqrstuvwxyz	<code>\$\mathbfup{abcdefghijklmnopqrstuvwxyz}\$ \\\</code>
ΑΒΓΔΕΖΗΘΙΚΑΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ	<code>\$\mathbfup{ }\$ \\\</code>
αβγδεζηθικλμνξοπρστυφχψω εθκφρπ	<code>\$\mathbfup{ }\$ \\\</code>

```

1150 \cs_new:Npn \um_config_mathbfup: {
1151   \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnum}
1152   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLatin}
1153   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplatin}
1154   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGreek}
1155   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgreek}
1156   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bfupLatin}{\um@usv@bfupLatin}
1157   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bfuplatin}{\um@usv@bfuplatin}
1158   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfupGreek}{\um@usv@bfupGreek}
1159   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfupgreek}{\um@usv@bfupgreek}
1160   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarTheta}
1161   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}
1162   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartial}
1163   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
1164   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1165   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1166   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1167   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1168   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1169 }

```

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

***ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz***

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

```

1170 \cs_new:Npn \um_config_mathbffrak: {
1171   \um_set_mathalphabet_numbers:Nnn{\mathbffrak}{\um@usv@num}{\um@usv@bfnum}
1172   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin,\um@usv@itLatin,\um@usv@frakLatin}
1173   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin,\um@usv@fraklatin}
1174 }

```

7.2.5 Bold script or calligraphic: `\mathbfscr`

***ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz***

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

```

1175 \cs_new:Npn \um_config_mathbfscr: {
1176   \um_set_mathalphabet_numbers:Nnn{\mathbfscr}{\um@usv@num}{\um@usv@bfnum}
1177   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrLatin}
1178   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrLatin}
1179 }

```

7.2.6 Bold sans serif: `\mathbfsf`

0123456789
 ABCDEFGHIJKLMNOPQRSTUVWXYZ
 abcdefghijklmnopqrstuvwxyz
 ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ
 αβγδεζηθικλμνξοπρστυφχψω εδκφρω

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

These use the sans-style settings rather than bold-style.
 Numbers (always upright) and letters:

```
1180 \cs_new:Npn \um_config_mathbfsf: {
1181   \bool_if:NTF \g_um_sfliteral_bool {
1182     \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1183     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin}{\um@usv@bfsfupLatin}
1184     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@uplatin}{\um@usv@bfsfuplatin}
1185     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itLatin}{\um@usv@bfsfitLatin}
1186     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itlatin}{\um@usv@bfsfitlatin}
1187     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek}{\um@usv@bfsfupGreek}
1188     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek}{\um@usv@bfsfupgreek}
1189     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itGreek}{\um@usv@bfsfitGreek}
1190     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itgreek}{\um@usv@bfsfitgreek}
1191     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{1D767}
1192     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla}{1D76F}
1193     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial}{1D789}
1194     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon}{1D78A}
1195     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta}{1D78B}
1196     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa}{1D78C}
1197     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi}{1D78D}
1198     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho}{1D78E}
1199     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi}{1D78F}
1200     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{1D7A1}
1201     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itNabla}{\um@usv@bfsfitNabla}
1202     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itpartial}{\um@usv@bfsfitpartial}
1203     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarepsilon}{1D7C4}
1204     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvartheta}{1D7C5}
1205     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarkappa}{1D7C6}
1206     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarphi}{1D7C7}
1207     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarrho}{1D7C8}
1208     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarpi}{1D7C9}
1209   }{
1210     \bool_if:NTF \g_um_upsans_bool {
1211       \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1212       \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfup}
1213       \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfup}
```



```

1214 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfup
1215 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfup
1216 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfsfup
1217 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfup
1218 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsfup
1219 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfsfup
1220 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfsfup
1221 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfsfup
1222 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfsfup
1223 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfsfup
1224 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfsfup
1225 }{
1226 \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@enum}{\um@usv@bfsfnum}
1227 \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfup
1228 \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfup
1229 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfup
1230 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfup
1231 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{\um@usv@bfsfup
1232 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfup
1233 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsfup
1234 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfsfup
1235 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfsfup
1236 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfsfup
1237 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfsfup
1238 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfsfup
1239 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfsfup }
1240 }
1241 }

```

7.2.7 Bold upright sans serif: `\mathbfsfup`

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ
αβγδεζηθικλμνξοπρστυφχψω εδκφρς

```

\setmathfont{STIXGeneral-Bold}
$\mathbfsfup{0123456789}$ \l
$\mathbfsfup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \l
$\mathbfsfup{abcdefghijklmnopqrstuvwxyz}$ \l
$\mathbfsfup{ }$ \quad
$\mathbfsfup{ }$ \quad
$\mathbfsfup{ }$ \quad

```

Numbers (always upright) and letters:

```

1242 \cs_new:Npn \um_config_mathbfsfup: {
1243 \um_set_mathalphabet_numbers:Nnn{\mathbfsfup}{\um@usv@enum}{\um@usv@bfsfnum}
1244 \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfupL
1245 \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfupl
1246 \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfupG
1247 \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfupg

```

Others:

```

1248 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varTheta,\um@usv@itvarTheta}{1D767}
1249 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@Nabla,\um@usv@itNabla}{1D76F}
1250 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@partial,\um@usv@itpartial}{1D789}
1251 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D78A}
1252 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1253 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1254 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1255 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varrho,\um@usv@itvarrho}{1D78E}
1256 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1257 }

```

7.2.8 Bold italic sans serif: `\mathbfsfit`

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ Θ
αβγδεζηθικλμνξοπρστυφχψω εδκφρω

```

\setmathfont{STIXGeneral-BoldItalic}
$\mathbfsfit{0123456789}$ \\\
$\mathbfsfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\\
$\mathbfsfit{abcdefghijklmnopqrstuvwxyz}$ \\\
$\mathbfsfit{ }$ \quad
$\mathbfsfit{ }$ \\\
$\mathbfsfit{ }$ \quad
$\mathbfsfit{ }$ \\\

```

```

1258 \cs_new:Npn \um_config_mathbfsfit: {
1259 \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@enum}{\um@usv@bfsfnum}
1260 \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfitL}
1261 \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfitl}
1262 \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfitG}
1263 \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfitg}

```

Other symbols:

```

1264 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{1D7A1}
1265 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNabla}
1266 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsfitp}
1267 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7C4}
1268 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{1D7C5}
1269 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{1D7C6}
1270 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{1D7C7}
1271 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{1D7C8}
1272 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{1D7C9}
1273 }

```

7.3 Definitions of the math symbols

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

`\um@scancharlet` We need to do some trickery to transform the `\UnicodeMathSymbol` argument
`\um@scanactivedef` "ABCDEF into the \TeX ‘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.

```
1274 \begingroup
1275   \char_make_other:N \^
1276   \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1277     \lowercase{
1278       \scantokens{\global\let#1=^^^^^#2}
1279     }
1280   }
```

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

```
1281   \gdef\um@scanactivedef"#1\@nil#2{
1282     \lowercase{
1283       \tl_rescan:nn{
1284         \ExplSyntaxOn
1285         \char_make_math_superscript:N\^
1286       }{
1287         \global\def^^^^^#1{#2}
1288       }
1289     }
1290   }
1291 \endgroup
```

Now give `\UnicodeMathSymbol` a definition in terms of `\um@scancharlet` and
we’re good to go. Make sure `#` is an ‘other’ so that we don’t get confused with
`\mathoctothorpe`.

```
1292 \begingroup
1293   \def\UnicodeMathSymbol#1#2#3#4{
1294     \um@scancharlet#2=#1\@nil
1295   }
1296   \char_make_other:N \#
1297   \@input{unicode-math-table.tex}
1298 \endgroup
```

Fix `\backslash`:

```
1299 \group_begin:
1300   \lccode`\*='\
1301   \char_make_escape:N \|
1302   \char_make_other:N \
1303   |lowercase{
1304 |group_end: |let|backslash=*
```

8 Epilogue

Lots of little things to tidy up.

8.0.1 Primes

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

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

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

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

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

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

- Prime encountered; pcount=1.
- Scan ahead; if prime: pcount:=pcount+1; repeat.
- If not prime, stop scanning.
- If pcount=1, \prime, end.
- If pcount=2, check \primedouble; if it exists, use it, end; if not, goto last step.
- Ditto pcount=3 & \primetripel.
- Ditto pcount=4 & \primequadruple.
- If pcount>4 or the glyph doesn’t exist, insert pcount \primes with \primekern between each.

```
1305 \muskip_new:N \g_um_primekern_muskip
1306 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1307 \num_new:N \l_um_primecount_num
```

```

1308 \cs_new:Nn \um_nprimes:n {
1309   ^{
1310     \primesingle
1311     \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1312   }
1313 }
1314 \cs_new:Nn \um_nprimes_select:n {
1315   \prg_case_int:nnn {#1}{
1316     {1} { ^{\primesingle} }
1317     {2} {
1318       \um_glyph_if_exist:nTF {"2033} { ^{\primedouble} } {\um_nprimes:n {#1}}
1319     }
1320     {3} {
1321       \um_glyph_if_exist:nTF {"2034} { ^{\primetriple} } {\um_nprimes:n {#1}}
1322     }
1323     {4} {
1324       \um_glyph_if_exist:nTF {"2057} { ^{\primequadruple} } {\um_nprimes:n {#1}}
1325     }
1326   }{
1327     \um_nprimes:n {#1}
1328   }
1329 }

```

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.

```

1330 \cs_new:Nn \um_scanprime: {
1331   \num_zero:N \l_um_primecount_num
1332   \um_scanprime_collect:
1333 }
1334 \cs_new:Nn \um_scanprime_collect: {
1335   \num_incr:N \l_um_primecount_num
1336   \peek_meaning_remove:NTF ' {
1337     \um_scanprime_collect:
1338   }{
1339     \peek_meaning_remove:NTF \um_scanprime: {
1340       \um_scanprime_collect:
1341     }{
1342       \peek_meaning_remove:NTF ^^^^2032 {
1343         \um_scanprime_collect:
1344       }{
1345         \um_nprimes_select:n {\l_um_primecount_num}
1346       }
1347     }
1348   }
1349 }

```

```

1350 \cs_set_eq:NN \prime \um_scanprime:
1351 \group_begin:
1352 \char_make_active:N \prime
1353 \char_make_active:n {"2032}
1354 \cs_gset_eq:NN \prime \um_scanprime:
1355 \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1356 \group_end:

```

8.0.2 Unicode radicals

Undo the damage made to `\sqrt`:

```

1357 \DeclareRobustCommand\sqrt{\@ifnextchar[\@sqrt\sqrtsign}

```

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

`#2` : Leading superscript for the sqrt sign

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

```

1358 \def\r@@t#1#2{
1359 \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
1360 \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1361 \raise \dimexpr(
1362 \um@fontdimen@percent{65}{\um@font}\ht\z@-
1363 \um@fontdimen@percent{65}{\um@font}\dp\z@
1364 )\relax
1365 \copy \rootbox
1366 \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1367 \box \z@
1368 }

```

8.0.3 Unicode sub- and super-scripts

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

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

First, the setup of each mathactive char:

```

1369 \prop_new:N \g_um_supers_prop
1370 \prop_new:N \g_um_subs_prop
1371 \cs_generate_variant:Nn \prop_gput:Nnn {Nxn}
1372 \cs_generate_variant:Nn \prop_get:NnN {cxN}
1373 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}

```

```

1374
1375 \group_begin:
1376
1377 % Populate a property list with superscript characters; their mean-
1378 % ing as their key,
1379 % for reasons that will become apparent soon, and their replace-
1380 % ment as each key's value.
1381 % Then make the superscript active and bind it to the scanning function.
1382 %
1383 % \cs{scantokens} makes this process much simpler since we can acti-
1384 % vate the char
1385 % and assign its meaning in one step.
1386 \cs_set:Nn \um_setup_active_superscript:nn {
1387   \prop_gput:Nxn \g_um_supers_prop {\meaning #1} {#2}
1388   \char_make_active:n {'#1}
1389   \global\XeTeXmathcodenum '#1 = "1FFFFF \scan_stop:
1390   \scantokens{
1391     \cs_gset:Npn #1 {
1392       \tl_set:Nn \l_um_ss_chain_tl {#2}
1393       \cs_set_eq:NN \um_sub_or_super:n \sp
1394       \tl_set:Nn \l_um_tmpa_tl {supers}
1395       \um_scan_sscript:
1396     }
1397   }
1398 }
1399
1400 \um_setup_active_superscript:nn {^^^2070} {0}
1401 \um_setup_active_superscript:nn {^^^00b9} {1}
1402 \um_setup_active_superscript:nn {^^^00b2} {2}
1403 \um_setup_active_superscript:nn {^^^00b3} {3}
1404 \um_setup_active_superscript:nn {^^^2074} {4}
1405 \um_setup_active_superscript:nn {^^^2075} {5}
1406 \um_setup_active_superscript:nn {^^^2076} {6}
1407 \um_setup_active_superscript:nn {^^^2077} {7}
1408 \um_setup_active_superscript:nn {^^^2078} {8}
1409 \um_setup_active_superscript:nn {^^^2079} {9}
1410 \um_setup_active_superscript:nn {^^^207a} {+}
1411 \um_setup_active_superscript:nn {^^^207b} {-}
1412 \um_setup_active_superscript:nn {^^^207c} {=}
1413 \um_setup_active_superscript:nn {^^^207d} {(}
1414 \um_setup_active_superscript:nn {^^^207e} {)}
1415 \um_setup_active_superscript:nn {^^^2071} {i}
1416 \um_setup_active_superscript:nn {^^^207f} {n}
1417
1418 % Ditto above.
1419 \cs_set:Nn \um_setup_active_subscript:nn {

```

```

1417 \prop_gput:Nxn \g_um_subs_prop    {\meaning #1} {#2}
1418 \char_make_active:n {'#1}
1419 \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1420 \scantokens{
1421   \cs_gset:Npn #1 {
1422     \tl_set:Nn \l_um_ss_chain_tl {#2}
1423     \cs_set_eq:NN \um_sub_or_super:n \sb
1424     \tl_set:Nn \l_um_tmpa_tl {subs}
1425     \um_scan_sscript:
1426   }
1427 }
1428 }
1429
1430 \um_setup_active_subscript:nn {^^^2080} {0}
1431 \um_setup_active_subscript:nn {^^^2081} {1}
1432 \um_setup_active_subscript:nn {^^^2082} {2}
1433 \um_setup_active_subscript:nn {^^^2083} {3}
1434 \um_setup_active_subscript:nn {^^^2084} {4}
1435 \um_setup_active_subscript:nn {^^^2085} {5}
1436 \um_setup_active_subscript:nn {^^^2086} {6}
1437 \um_setup_active_subscript:nn {^^^2087} {7}
1438 \um_setup_active_subscript:nn {^^^2088} {8}
1439 \um_setup_active_subscript:nn {^^^2089} {9}
1440 \um_setup_active_subscript:nn {^^^208a} {+}
1441 \um_setup_active_subscript:nn {^^^208b} {-}
1442 \um_setup_active_subscript:nn {^^^208c} {=}
1443 \um_setup_active_subscript:nn {^^^208d} {(}
1444 \um_setup_active_subscript:nn {^^^208e} {)}
1445 \um_setup_active_subscript:nn {^^^2090} {a}
1446 \um_setup_active_subscript:nn {^^^2091} {e}
1447 \um_setup_active_subscript:nn {^^^1d62} {i}
1448 \um_setup_active_subscript:nn {^^^2092} {o}
1449 \um_setup_active_subscript:nn {^^^1d63} {r}
1450 \um_setup_active_subscript:nn {^^^1d64} {u}
1451 \um_setup_active_subscript:nn {^^^1d65} {v}
1452 \um_setup_active_subscript:nn {^^^2093} {x}
1453 \um_setup_active_subscript:nn {^^^1d66} {\beta}
1454 \um_setup_active_subscript:nn {^^^1d67} {\gamma}
1455 \um_setup_active_subscript:nn {^^^1d68} {\rho}
1456 \um_setup_active_subscript:nn {^^^1d69} {\phi}
1457 \um_setup_active_subscript:nn {^^^1d6a} {\chi}
1458
1459 \group_end:
1460
1461 % The scanning command, evident in its purpose:
1462 \cs_new:Nn \um_scan_sscript: {

```



```

1463 \um_scan_sscript:TF {
1464   \um_scan_sscript:
1465   }{
1466     \um_sub_or_super:n {\l_um_ss_chain_tl}
1467   }
1468 }
1469
1470 % The main theme here is stolen from the source to the various \cs{peek_} func-
1471   tions.
1472 % Consider this function as simply boilerplate:
1473 \cs_new:Nn \um_scan_sscript:TF {
1474   \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
1475   \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
1476   \tl_set:Nx \l_peek_false_tl { \exp_not:n{\group_align_safe_end: #2}}
1477   \group_align_safe_begin:
1478   \peek_after:NN \um_peek_execute_branches_ss:
1479 }
1480
1481 % We do not skip spaces when scanning ahead, and we explicitly wish to
1482 % bail out on encountering a space or a brace.
1483 \cs_new:Npn \um_peek_execute_branches_ss: {
1484   \bool_if:nTF {
1485     \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
1486     \token_if_eq_catcode_p:NN \l_peek_token \c_group_end_token ||
1487     \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
1488   } { \l_peek_false_tl }
1489   { \um_peek_execute_branches_ss_aux: }
1490 }
1491
1492 % This is the actual comparison code.
1493 % Because the peeking has already tokenised the next token,
1494 % it's too late to extract its charcode directly. Instead,
1495 % we look at its meaning, which remains a 'character' even
1496 % though it is itself math-active. If the character is ever
1497 % made fully active, this will break our assumptions!
1498 %
1499 % If the char's meaning exists as a property list key, we
1500 % build up a chain of sub-/superscripts and iterate. (If not, exit and
1501 % typeset what we've already collected.)
1502 \cs_new:Nn \um_peek_execute_branches_ss_aux: {
1503   \prop_if_in:cxTF
1504     {g_um_\l_um_tmpa_tl _prop}
1505     {\meaning\l_peek_token}
1506   {
1507     \prop_get:cxN

```

```

1508     {g_um_\l_um_tmpa_tl _prop}
1509     {\meaning\l_peek_token}
1510     \l_um_tmpb_tl
1511     \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmpb_tl
1512     \l_peek_true_tl
1513   }
1514   {\l_peek_false_tl}
1515 }

```

8.0.4 Synonyms and all the rest

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

```

1516 \def\operator@font{\um_setup_mathup:}
1517 \def\to{\rightarrow}
1518 \def\vec{\overrightarrow}
1519 \def\le{\leq}
1520 \def\ge{\geq}
1521 \def\neq{\neq}

```

Define `\colon` as a `mathpunct` ':'. This is wrong: it should be U+003A: COLON instead!

```

1522 \@ifpackageloaded{amsmath}{
1523   % define their own colon, perhaps I should just steal it.
1524 }{
1525   \cs_set_protected:Npn \colon {
1526     \bool_if:NTF \g_um_literal_colon_bool {:} { \mathpunct{:} }
1527   }
1528 }

```

`\mathcal`

```

1529 \def\mathcal{\mathscr}

```

`\mathrm`

```

1530 \def\mathrm{\mathup}

```

8.0.5 Compatibility

Note that `amsmath` will always be loaded before `unicode-math`. (Conflicts occur if you try it the other way around.)

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

```

1531 \@ifpackageloaded{amsmath}{
1532   \tl_remove_in:Nn \@begindocumenthook {

```

```

1533         \mathchardef\std@minus\mathcode`\-\relax
1534         \mathchardef\std@equal\mathcode`\=\relax
1535     }
1536 }{}

```

- This code is to improve the output of alphabetic symbols in text of operator names (`\sin`, `\cos`, etc.). Just comment out the offending lines for now:

```

1537     \ifpackageloaded{amsopn}{
1538         \cs_set:Npn \newmcodes@ {
1539             \mathcode`\'39
1540             \mathcode`\*42
1541             \mathcode`\."613A%
1542         % \ifnum\mathcode`\-45 \else
1543         %     \mathchardef\std@minus\mathcode`\-\relax
1544         % \fi
1545             \mathcode`\-45
1546             \mathcode`\ /47
1547             \mathcode`\:"603A\relax
1548         }
1549     }{}

```

Octothorpe is an odd one:

```

1550 \AtBeginDocument{
1551     \def\#{\mode_if_math:TF{\mathoctothorpe}{\char`\#}}
1552 }

```

Overriding amsmath definitions:

```

1553 \AtBeginDocument{
1554     \def\@cdots{\mathinner{\cdots}}
1555 }

```

Interaction with beamer:

```

1556 \ifclassloaded{beamer}{
1557     \ifbeamer@suppressreplacements\else
1558         \PackageWarningNoLine{unicode-math}{
1559             Disabling~ beamer's~ math~ setup.^J
1560             Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1561         }
1562         \beamer@suppressreplacementstrue
1563     \fi
1564 }{}

```

The end.

```

1565 \ExplSyntaxOff

```

File II

STIX table data extraction

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

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

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

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

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

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

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

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

Print the actual entry corresponding to the unicode character:

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

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

```

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}/ ' \

```

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

A.1 Overview

Maths symbol fonts Fonts for symbols: , ,

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

 $\backslash\mathrm{DeclareMathAlphabet}\{cmd\}NFSS\,decl.$
$$\backslash\text{DeclareSymbolFontAlphabet}\{ \textit{cmd} \}\{ \textit{name} \}$$

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

$$\backslash\mathrm{SetSymbolFont}\{ name \}\{ maths\ version \} NFSS\ decl.$$
$$\backslash\mathrm{SetMathAlphabet}\{cmd\}\{maths\ version\}NFSS\ decl.$$

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

Delimiters and radicals use wrappers around \TeX 's `\delimiter`/`\radical` primitives, which are re-designed in \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 \XTeX .

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

\XTeX math font dimensions

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

\fontdimen	Dimension name	Description
10	SCRIPTPERCENTSCALEDOWN	Percentage of scaling down for script level 1. Suggested value: 80%.
11	SCRIPTSCRIPTPERCENTSCALEDOWN	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	DELIMITEDSUBFORMULAMINHEIGHT	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height \times 1.5.
13	DISPLAYOPERATORMINHEIGHT	Minimum height of n-ary operators (such as integral and summation) for formulas in display mode.
14	MATHLEADING	White space to be left between math formulas to ensure proper line spacing. For example, for applications that treat line gap as a part of line ascender, formulas with ink going above (os2.sTypoAscender + os2.sTypoLineGap – MathLeading) or with ink going below os2.sTypoDescender will result in increasing line height.
15	AXISHEIGHT	Axis height of the font.
16	ACCENTBASEHEIGHT	Maximum (ink) height of accent base that does not require raising the accents. Suggested: x-height of the font (os2.sxHeight) plus any possible overshoots.
17	FLATTENEDACCENTBASEHEIGHT	Maximum (ink) height of accent base that does not require flattening the accents. Suggested: cap height of the font (os2.sCapHeight).
18	SUBSCRIPTSHIFTDOWN	The standard shift down applied to subscript elements. Positive for moving in the downward direction. Suggested: os2.ySubscriptYOffset.
19	SUBSCRIPTTOPMAX	Maximum allowed height of the (ink) top of subscripts that does not require moving subscripts further down. Suggested: $/5$ x-height.

\fontdimen	Dimension name	Description
20	SUBSCRIPTBASELINEDROPMIN	Minimum allowed drop of the baseline of subscripts relative to the (ink) bottom of the base. Checked for bases that are treated as a box or extended shape. Positive for subscript baseline dropped below the base bottom.
21	SUPERSCRIPSHIFTUP	Standard shift up applied to superscript elements. Suggested: $os2.ySuperscriptYOffset$.
22	SUPERSCRIPSHIFTUPCRAMPED	Standard shift of superscripts relative to the base, in cramped style.
23	SUPERSCRIPBOTTOMMIN	Minimum allowed height of the (ink) bottom of superscripts that does not require moving subscripts further up. Suggested: $\frac{1}{4}$ x-height.
24	SUPERSCRIPBASELINEDROP-MAX	Maximum allowed drop of the baseline of superscripts relative to the (ink) top of the base. Checked for bases that are treated as a box or extended shape. Positive for superscript baseline below the base top.
25	SUBSUPERSCRIPGAPMIN	Minimum gap between the superscript and subscript ink. Suggested: $4 \times$ default rule thickness.
26	SUPERSCRIPBOTTOMMAX-WITHSUBSCRIPT	The maximum level to which the (ink) bottom of superscript can be pushed to increase the gap between superscript and subscript, before subscript starts being moved down. Suggested: $\frac{1}{5}$ x-height.
27	SPACEAFTERSCRIP	Extra white space to be added after each subscript and superscript. Suggested: 0.5pt for a 12 pt font.
28	UPPERLIMITGAPMIN	Minimum gap between the (ink) bottom of the upper limit, and the (ink) top of the base operator.
29	UPPERLIMITBASELINERISEMIN	Minimum distance between baseline of upper limit and (ink) top of the base operator.
30	LOWERLIMITGAPMIN	Minimum gap between (ink) top of the lower limit, and (ink) bottom of the base operator.

\fontdimen	Dimension name	Description
31	LOWERLIMITBASELINEDROP-MIN	Minimum distance between baseline of the lower limit and (ink) bottom of the base operator.
32	STACKTOPSHIFTUP	Standard shift up applied to the top element of a stack.
33	STACKTOPDISPLAYSTYLESHIFT-UP	Standard shift up applied to the top element of a stack in display style.
34	STACKBOTTOMSHIFTDOWN	Standard shift down applied to the bottom element of a stack. Positive for moving in the downward direction.
35	STACKBOTTOMDISPLAYSTYLE-SHIFTDOWN	Standard shift down applied to the bottom element of a stack in display style. Positive for moving in the downward direction.
36	STACKGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element. Suggested: 3×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	STRETCHSTACKBOTTOMSHIFT-DOWN	Standard shift down applied to the bottom element of the stretch stack. Positive for moving in the downward direction.
40	STRETCHSTACKGAPABOVEMIN	Minimum gap between the ink of the stretched element, and the (ink) bottom of the element above. Suggested: UpperLimitGapMin
41	STRETCHSTACKGAPBELOWMIN	Minimum gap between the ink of the stretched element, and the (ink) top of the element below. Suggested: LowerLimitGapMin.
42	FRACTIONNUMERATORSHIFTUP	Standard shift up applied to the numerator.
43	FRACTIONNUMERATOR-DISPLAYSTYLESHIFTUP	Standard shift up applied to the numerator in display style. Suggested: StackTopDisplayStyleShiftUp.

\fontdimen	Dimension name	Description
44	FRACTIONDENOMINATORSHIFT-DOWN	Standard shift down applied to the denominator. Positive for moving in the downward direction.
45	FRACTIONDENOMINATOR-DISPLAYSTYLESHIFTDOWN	Standard shift down applied to the denominator in display style. Positive for moving in the downward direction. Suggested: StackBottomDisplayStyleShiftDown.
46	FRACTIONNUMERATORGAP-MIN	Minimum tolerated gap between the (ink) bottom of the numerator and the ink of the fraction bar. Suggested: default rule thickness
47	FRACTIONNUMDISPLAYSTYLE-GAPMIN	Minimum tolerated gap between the (ink) bottom of the numerator and the ink of the fraction bar in display style. Suggested: 3×default rule thickness.
48	FRACTIONRULETHICKNESS	Thickness of the fraction bar. Suggested: default rule thickness.
49	FRACTIONDENOMINATORGAP-MIN	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar. Suggested: default rule thickness
50	FRACTIONDENOMDISPLAY-STYLEGAPMIN	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar in display style. Suggested: 3×default rule thickness.
51	SKEWEDFRACTION-HORIZONTALGAP	Horizontal distance between the top and bottom elements of a skewed fraction.
52	SKEWEDFRACTIONVERTICAL-GAP	Vertical distance between the ink of the top and bottom elements of a skewed fraction.
53	OVERBARVERTICALGAP	Distance between the overbar and the (ink) top of the base. Suggested: 3×default rule thickness.
54	OVERBARRULETHICKNESS	Thickness of overbar. Suggested: default rule thickness.
55	OVERBAREXTRAASCENDER	Extra white space reserved above the overbar. Suggested: default rule thickness.

\fontdimen	Dimension name	Description
56	UNDERBARVERTICALGAP	Distance between underbar and (ink) bottom of the base. Suggested: 3×default rule thickness.
57	UNDERBARRULETHICKNESS	Thickness of underbar. Suggested: default rule thickness.
58	UNDERBAREXTRADESCENDER	Extra white space reserved below the underbar. Always positive. Suggested: default rule thickness.
59	RADICALVERTICALGAP	Space between the (ink) top of the expression and the bar over it. Suggested: 1¼ default rule thickness.
60	RADICALDISPLAYSTYLE- VERTICALGAP	Space between the (ink) top of the expression and the bar over it. Suggested: default rule thickness + ¼ x-height.
61	RADICALRULETHICKNESS	Thickness of the radical rule. This is the thickness of the rule in designed or constructed radical signs. Suggested: default rule thickness.
62	RADICALEXTRAASCENDER	Extra white space reserved above the radical. Suggested: RadicalRuleThickness.
63	RADICALKERNBEFOREDEGREE	Extra horizontal kern before the degree of a radical, if such is present. Suggested: 5/18 of em.
64	RADICALKERNAFTERDEGREE	Negative kern after the degree of a radical, if such is present. Suggested: −10/18 of em.
65	RADICALDEGREEBOTTOM- RAISEPERCENT	Height of the bottom of the radical degree, if such is present, in proportion to the ascender of the radical sign. Suggested: 60%.

File IV

Some manner of unit testing

B The bold alphabets

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

```

1
2 \documentclass{article}
3 \usepackage[a6paper]{geometry}
4 \usepackage{fontspec}
5 \setmainfont{TeX Gyre Pagella}
6 \usepackage{unicode-math}
7 \def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
8 \def\uplatin{abcdefghijklmnopqrstuvwxyz}
9 \def\upGreek{
10 \def\upgreek{
11 \def\itLatin{
12 \def\itlatin{
13 \def\itGreek{
14 \def\itgreek{
15 \def\bfupLatin{
16 \def\bfuplatin{
17 \def\bfupGreek{
18 \def\bfupgreek{
19 \def\bfItLatin{
20 \def\bfItlatin{
21 \def\bfItGreek{
22 \def\bfItgreek{
23 \providecommand\mathalphabet{\mathbf}
24 \def\testmath#1{%
25   \makebox[\linewidth][l]{%
26     \makebox[0pt][l]{\mathalphabet{\csname up#1\endcsname}}}%
27     \makebox[0pt][l]{\mathalphabet{\csname it#1\endcsname}}}%
28     \makebox[0pt][l]{\csname bfup#1\endcsname}}}%
29     \makebox[0pt][l]{\csname bfIt#1\endcsname}}}%
30   }}
31 \begin{document}
32 \setmathfont[Colour=2255FF55]{Cambria Math}
33 \parindent=0pt
34 \voffset=-1in
35 \hoffset=-1in
36 \setbox0=\vbox{%
37   \testmath{Latin}\\
38   \testmath{latin}\\
39   \testmath{Greek}\\
40   \testmath{greek}}
41 \dimen0=\ht0
42 \advance\dimen0\dp0
43 \edef\papersize{papersize=\the\wd0,\the\dimen0}
44 \setbox255=\vbox{\special{\papersize}\box0}
45 \shipout\box255
46 \end{document}

```


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
\#	1296, 1551
\'	1352, 1539
*	447, 1300, 1540
\-	446, 1533, 1542, 1543, 1545
\.	1541
\/	1546
\:	449, 1547
\:::	703
\::f	703
\::n	703
\=	1534
\DeclareMathDelimiter	295
\DeclareMathSizes	260
\backslashchar	893, 912
\begindocumenthook	1532
\ecclvi	320
\ecdots	1554
\elt	568–572, 575, 579, 581
\empty	371,
	372, 409, 496, 619, 626, 641, 662, 667
\ifclassloaded	1556
\ifnextchar	1357
\ifpackageloaded	1522, 1531, 1537
\ii	625, 626, 628, 630, 633, 638
\input	426, 1297
\marker	638, 657
\nil	329,
	477, 638, 651–654, 1276, 1281, 1294
\preamblecmds	304
\sqrt	1357
\tempa	120, 167, 193, 202, 220, 238,
	246, 382, 398, 617, 629, 653, 655, 667
\tempb	120, 121, 167, 168,
	193, 194, 202, 203, 220, 221, 238,
	239, 246, 247, 617, 618, 656, 657, 662
\tempswafalse	627
\tempswattrue	631, 634, 659, 664, 669, 674
\um@bfliteraltrue	162, 187
\um@bfupGreekfalse	126, 169
\um@bfupGreektrue	139, 152, 175, 181
\um@bfupLatinfalse	129, 172
\um@bfupLatintrue	142, 155, 178, 184
\um@bfupgreekfalse	127, 140, 170, 176
\um@bfupgreektrue	153, 182
\um@bfuplatinfalse	130, 173
\um@bfuplatintrue	143, 156, 179, 185
\um@fontspec@featuretrue	373
\um@literaltrue	161
\um@ot@math@true	401
\um@upGreekfalse	122
\um@upGreektrue	135, 148
\um@upLatinfalse	124, 137
\um@upLatintrue	150
\um@upNablafalse	131, 206
\um@upNablatrie	144, 157, 204
\um@upgreekfalse	123, 136
\um@upgreektrue	149
\um@uplatinfalse	125, 138, 151
\um@uppartialfalse	
	128, 141, 171, 177, 224
\um@uppartialtrue	154, 183, 222
\xDeclareMathDelimiter	296
\xxDeclareMathDelimiter	294
\	10–13, 17, 23–33, 37–39, 1300, 1302
\^	33, 1275, 1285
\	1301
Numbers	
\0	23
_	17–20, 23–32
A	
\A	24
\a	25
\addnolimits	<u>574</u>
\addtoversion	<u>267</u>
\advance	42

<code>\alloc@</code>	320	<code>\char_make_escape:N</code>	1301
<code>\Alpha</code>	730	<code>\char_make_math_superscript:N</code> ..	1285
<code>\alpha</code>	755	<code>\char_make_other:N</code> ...	1275, 1296, 1302
<code>\alpha@elt</code>	271	<code>\chardef</code>	320
<code>\alpha@list</code>	270	<code>\Chi</code>	752
<code>\AtBeginDocument</code>	728, 1550, 1553	<code>\chi</code>	781, 1457
<code>\awint</code>	570	<code>\cirfnint</code>	570
B		<code>\clist_map_break:</code>	884
<code>\B</code>	993	<code>\clist_map_inline:Nn</code>	553
<code>\beamer@suppressreplacementstrue</code>	1562	<code>\clist_map_inline:nn</code>	482, 548, 555, 880, 887
<code>\begin</code>	31	<code>\clist_map_variable:NNn</code>	625
<code>\begingroup</code>	326, 578, 1274, 1292	<code>\clist_map_variable:nNn</code> ..	682, 705, 711
<code>\Beta</code>	731	<code>\colon</code>	1525
<code>\beta</code>	756, 1453	<code>\copy</code>	1365
<code>\bfitGreek</code>	21	<code>\cs</code>	1381, 1470
<code>\bfitgreek</code>	22	<code>\cs_generate_variant:Nn</code> ...	1371–1373
<code>\bfitLatin</code>	19	<code>\cs_gset:cpn</code>	334, 344
<code>\bfitlatin</code>	20	<code>\cs_gset:Npn</code> ..	347, 352, 1276, 1388, 1421
<code>\bfupGreek</code>	17	<code>\cs_gset:Npx</code>	357
<code>\bfupgreek</code>	18	<code>\cs_gset_eq:NN</code>	1354, 1355
<code>\bfupLatin</code>	15	<code>\cs_if_exist:cF</code>	928
<code>\bfuplatin</code>	16	<code>\cs_if_exist:cT</code>	913
<code>\bgroup</code>	933	<code>\cs_new:Nn</code>	433, 445, 476, 481, 486, 489, 704, 710, 719, 722, 725, 860, 879, 905, 927, 1308, 1314, 1330, 1334, 1462, 1472, 1502
<code>\bool_if:NF</code>	448	<code>\cs_new:Npn</code> 942, 945, 948, 953, 963, 979, 991, 1002, 1008, 1016, 1019, 1038, 1043, 1048, 1053, 1130, 1150, 1170, 1175, 1180, 1242, 1258, 1482	
<code>\bool_if:NTF</code>	760, 779, 785, 790, 1020, 1027, 1181, 1210, 1526	<code>\cs_set:cpn</code>	929
<code>\bool_if:nTF</code>	1483	<code>\cs_set:Nn</code>	209, 227, 366, 437, 440, 495, 547, 552, 681, 691, 694, 697, 700, 795, 806, 816, 823, 832, 841, 918, 921, 1383, 1416
<code>\bool_new:N</code>	22, 190, 191, 245	<code>\cs_set:Npn</code>	703, 1538
<code>\bool_set_false:N</code>	132, 133, 159, 164, 195, 240, 250	<code>\cs_set_eq:cN</code>	922
<code>\bool_set_true:N</code>	145, 146, 158, 163, 197, 199, 242, 248	<code>\cs_set_eq:NN</code>	412–415, 419–422, 1350, 1390, 1423
<code>\box</code>	44, 45, 1367	<code>\cs_set_protected:cpn</code>	932
C		<code>\cs_set_protected:Npn</code>	1525
<code>\C</code>	982, 1010	<code>\cs_to_str:N</code>	329, 334, 344, 563
<code>\c_group_begin_token</code>	1484	<code>\csname</code> ...	26–29, 323, 329, 332, 335, 367, 374, 491, 563, 881, 888, 906, 936
<code>\c_group_end_token</code>	1485		
<code>\c_peek_true_remove_next_tl</code>	1474		
<code>\c_space_token</code>	1486		
<code>\cdots</code>	1554		
<code>\cdp@elt</code>	258		
<code>\cdp@list</code>	257		
<code>\char</code>	1551		
<code>\char_make_active:N</code>	1352		
<code>\char_make_active:n</code> 327, 1353, 1385, 1418			

D	
<code>\DeclareDocumentCommand</code>	369
<code>\DeclareMathAccent</code>	288
<code>\DeclareMathAlphabet</code>	284
<code>\DeclareMathDelimiter</code>	293
<code>\DeclareMathRadical</code>	299
<code>\DeclareMathSizes</code>	259
<code>\DeclareMathSymbol</code>	290
<code>\DeclareMathVersion</code>	264, 376
<code>\DeclareRobustCommand</code>	1357
<code>\DeclareSymbolFont</code>	277, 424
<code>\DeclareSymbolFontAlphabet</code>	301
<code>\DeclareSymbolFontAlphabet@</code>	302
<code>\def</code>	7–118, 306, 309, 320, 322, 324, 375, 567, 575, 577, 579, 586, 588, 651, 653–656, 730–759, 762–778, 781–784, 787–789, 792, 793, 1287, 1293, 1358, 1516–1521, 1529, 1530, 1551, 1554
<code>\define@choicekey</code>	119, 167, 192, 202, 220, 238, 246, 617
<code>\define@cmdkey</code>	613–616
<code>\define@key</code>	596
<code>\define@mathalphabet</code>	265
<code>\define@mathgroup</code>	266
<code>\Delta</code>	733
<code>\delta</code>	758
<code>\dimen</code>	41–43
<code>\dimexpr</code>	307, 400, 1361
<code>\do</code>	304
<code>\documentclass</code>	2
<code>\dorestore@version</code>	274
<code>\dp</code>	42, 1363
E	
<code>\E</code>	994
<code>\e</code>	1004
<code>\edef</code>	43, 382, 652, 653
<code>\egroup</code>	930, 935
<code>\else</code>	214, 232, 312, 315, 341, 345, 350, 355, 358, 402, 416, 456, 469, 500, 508, 511, 516, 522, 534, 544, 580, 599, 632, 637, 643, 661, 666, 671, 819, 827, 836, 850, 1087, 1090, 1096, 1103, 1116, 1542, 1557
<code>\else:</code>	925
<code>\encodingdefault</code>	425, 919
<code>\end</code>	46
<code>\endcsname</code>	26–29, 323, 329, 332, 335, 367, 374, 491, 563, 881, 888, 906, 936
<code>\endgroup</code>	330, 584, 1291, 1298
<code>\Epsilon</code>	734
<code>\epsilon</code>	759
<code>\Eta</code>	736
<code>\eta</code>	763
<code>\etex_iffontchar:D</code>	925
<code>\ExecuteOptionsX</code>	253
<code>\exp_args:Nnff</code>	703, 706, 713
<code>\exp_args:No</code>	549, 556
<code>\exp_not:n</code>	1473, 1475
<code>\expandafter</code>	323, 331, 336, 338, 342, 562, 575, 628, 630, 633, 638, 652, 653, 657, 935, 936
<code>\ExplSyntaxOff</code>	1565
<code>\ExplSyntaxOn</code>	6, 1284
F	
<code>\F</code>	995
<code>\f@size</code>	374
<code>\fi</code>	165, 188, 200, 207, 218, 225, 236, 243, 251, 317, 318, 339, 360–364, 408, 423, 459, 474, 505, 513, 518, 525, 542, 543, 545, 558, 582, 604, 620, 635, 636, 639, 645, 647, 648, 660, 665, 670, 675–679, 821, 830, 839, 858, 1092, 1099, 1106, 1125, 1128, 1544, 1563
<code>\fi:</code>	925
<code>\fint</code>	570
<code>\font</code>	399
<code>\fontdimen</code>	307, 400, 1360, 1366
<code>\fontname</code>	896, 912
G	
<code>\g</code>	1005
<code>\g@addto@macro</code>	562, 642, 644
<code>\g_um_literal_colon_bool</code>	245, 248, 250, 448, 1526
<code>\g_um_math_alphabet_name_greek_tl</code>	903
<code>\g_um_math_alphabet_name_greek_tl</code>	902
<code>\g_um_math_alphabet_name_Latin_tl</code>	901
<code>\g_um_math_alphabet_name_latin_tl</code>	900
<code>\g_um_math_alphabet_name_num_tl</code>	904
<code>\g_um_primekern_muskip</code>	1305, 1306, 1311

\g_um_sfliteral_bool	163, 191, 199, 1020, 1181
\g_um_subs_prop	1370, 1417
\g_um_supers_prop	1369, 1384
\g_um_texgreek_bool ..	22, 133, 146, 159, 164, 240, 242, 760, 779, 785, 790
\g_um_upsans_bool	132, 145, 158, 190, 195, 197, 1027, 1210
\Gamma	732
\gamma	757, 1454
\gdef	1281
\ge	1520
\geq	1520
\get@cdp	283
\glb@currsz	370
\global	328, 331, 348, 349, 353, 354, 359, 619, 1278, 1287, 1386, 1419
\group@elt	279
\group@list	278
\group_align_safe_begin:	1476
\group_align_safe_end:	1475
\group_begin:	1299, 1351, 1375
\group_end:	1356, 1459
H	
\H	983, 996, 1011
\h	826, 829, 966
\hbox	1359
\hoffset	35
\ht	41, 1362
I	
\I	997, 1012
\if@tempwa	640
\if@um@bfliteral	15, 460, 506, 1057
\if@um@bfupGreek	16, 519, 1100
\if@um@bfupgreek	17, 526, 1107
\if@um@bfupLatin	18, 509, 1088
\if@um@bfuplatin	19, 514, 1093
\if@um@fontspec@feature	8, 597
\if@um@literal	10, 451, 498
\if@um@ot@math@	9
\if@um@upGreek	11, 833
\if@um@upgreek	12, 842
\if@um@upLatin	13, 817
\if@um@uplatin	14, 824
\if@um@upNabla	20, 210
\if@um@uppartial	21, 228
\ifbeamer@suppressreplacements ..	1557
\ifcase 121, 168, 194, 203, 221, 239, 247, 618	
\ifdim	400
\ifin@	337, 343
\ifnum ..	554, 658, 663, 668, 672, 673, 1542
\ifx	310, 313, 325, 346, 351, 356, 409, 496, 580, 626, 629, 630, 633, 641, 657, 662, 667
\iiiint	568
\iiint	568
\iint	568
\in@	336, 342
\init@restore@version	273
\int	568
\intBar	570
\intbar	570
\intcap	572
\intclockwise	569
\intcup	572
\intlarhk	571
\intx	571
\Iota	738
\iota	765
\itGreek	13
\itgreek	14
\itLatin	11
\itlatin	12
K	
\Kappa	739
\kappa	766
\kern	1360, 1366
L	
\L	998
\l_peek_false_tl	1475, 1488, 1514
\l_peek_token ...	1484–1486, 1505, 1509
\l_peek_true_aux_tl	1473
\l_peek_true_tl	1474, 1512
\l_um_inc_num	712, 714, 715
\l_um_incr_num	683, 685, 687
\l_um_input_num 682, 685, 705, 707, 711, 714	
\l_um_primecount_num	1307, 1331, 1335, 1345
\l_um_script_features_tl	377, 388
\l_um_script_font_tl	379, 387

<code>\l_um_ss_chain_tl</code>	1389, 1422, 1466, 1511	<code>\mathinner</code>	1554
<code>\l_um_sscript_features_tl</code>	378, 392	<code>\mathit</code>	964–977
<code>\l_um_sscript_font_tl</code>	380, 391	<code>\mathoctothorpe</code>	1551
<code>\l_um_tmpa_tl</code>	1391, 1424, 1504, 1508	<code>\mathop</code>	325
<code>\l_um_tmpb_tl</code>	1510, 1511	<code>\mathopen</code>	346, 588
<code>\Lambda</code>	740	<code>\mathord</code>	452–455, 457, 458, 461–468, 470–473, 487
<code>\lambda</code>	767	<code>\mathpunct</code>	1526
<code>\lccode</code>	1300	<code>\mathrel</code>	449
<code>\le</code>	1519	<code>\mathrm</code>	1530
<code>\left</code>	587	<code>\mathscr</code>	992–1000, 1003–1006, 1529
<code>\left@primitive</code>	587, 588	<code>\mathsf</code>	1021–1025, 1028–1030, 1032–1034
<code>\leq</code>	1519	<code>\mathsf fit</code>	1044–1046
<code>\let</code>	321, 370–372, 587, 619, 1278	<code>\mathsf up</code>	1039–1041
<code>\linewidth</code>	25	<code>\mathtt</code>	919, 1049–1051
<code>\lowercase</code>	1277, 1282	<code>\mathup</code>	943, 946, 949–951, 954–961, 1530
<code>\lowint</code>	572	<code>\mddefault</code>	425, 919
M			
<code>\M</code>	999	<code>\meaning</code>	1384, 1417, 1505, 1509
<code>\m@th</code>	1359	<code>\mitAlpha</code>	730
<code>\makebox</code>	25–29	<code>\mitalpha</code>	755
<code>\mathaccent</code>	356	<code>\mitBeta</code>	731
<code>\mathalpha</code>	564, 686	<code>\mitbeta</code>	756
<code>\mathalphabet</code>	23, 26, 27	<code>\mitChi</code>	752
<code>\mathbb</code>	980–989	<code>\mitchi</code>	781
<code>\mathbf</code>	23, 1054–1056, 1058–1086, 1089, 1091, 1094, 1095, 1097, 1098, 1101, 1102, 1104, 1105, 1108–1115, 1117–1124, 1126, 1127	<code>\mitDelta</code>	733
<code>\mathbffrac</code>	1171–1173	<code>\mitdelta</code>	758
<code>\mathbf fit</code>	1131–1148	<code>\mitEpsilon</code>	734
<code>\mathbf fscr</code>	1176–1178	<code>\mitepsilon</code>	760, 785
<code>\mathbfsf</code>	1182–1208, 1211–1224, 1226–1239	<code>\mitEta</code>	736
<code>\mathbfsf fit</code>	1259–1272	<code>\miteta</code>	763
<code>\mathbfsfup</code>	1243–1256	<code>\mitGamma</code>	732
<code>\mathbfup</code>	1151–1168	<code>\mitgamma</code>	757
<code>\mathbin</code>	446, 447	<code>\mitIota</code>	738
<code>\mathcal</code>	1529	<code>\mitiota</code>	765
<code>\mathchar@type</code>	300, 333, 347, 349, 352, 354, 357, 359, 367, 490	<code>\mitKappa</code>	739
<code>\mathchardef</code>	1533, 1534, 1543	<code>\mitkappa</code>	766
<code>\mathclose</code>	351	<code>\mitLambda</code>	740
<code>\mathcode</code>	328, 1533, 1534, 1539–1543, 1545–1547	<code>\mitlambda</code>	767
<code>\mathfrak</code>	1009–1014, 1017	<code>\mitMu</code>	741
<code>\mathgroup</code>	320	<code>\mitmu</code>	768
		<code>\mitNu</code>	742
		<code>\mitnu</code>	769
		<code>\mitOmega</code>	754
		<code>\mitomega</code>	783
		<code>\mitOmicron</code>	744
		<code>\mitomicron</code>	771

$\backslash\Theta$	737	$\um@usv@bfraklatin$	59, 1173
$\backslash\theta$	764	$\um@usv@bfitGreek$	56, 520, 523, 813, 1063, 1104, 1134, 1138
$\backslash\mathrm{theum@fam}$	418	$\um@usv@bfitgreek$	57, 527, 535, 814, 1065, 1117, 1135, 1139
$\backslash\mathrm{thinmuskip}$	1306	$\um@usv@bfith$	99, 1066, 1098
$\backslash\mathrm{tl_map_inline:nn}$	255	$\um@usv@bfitLatin$	54, 510, 512, 809, 1059, 1091, 1132, 1136
$\backslash\mathrm{tl_put_right:Nv}$	1511	$\um@usv@bfitlatin$	55, 515, 517, 810, 1061, 1097, 1133, 1137
$\backslash\mathrm{tl_remove_in:Nn}$	304, 1532	$\um@usv@bfitNabla$ 110, 216, 462, 470, 1079, 1141
$\backslash\mathrm{tl_rescan:nn}$	1283	$\um@usv@bfitpartial$	116, 234, 466, 472, 1080, 1124, 1142
$\backslash\mathrm{tl_set:Nn}$	211–213, 215–217, 229–231, 233–235, 377–380, 410, 900–904, 1389, 1391, 1422, 1424	$\um@usv@bfitvarepsilon$ 101, 528, 536, 1081, 1118, 1143
$\backslash\mathrm{tl_set:Nx}$	418, 1473, 1475	$\um@usv@bfitvarkappa$ 103, 530, 538, 1083, 1120, 1145
$\backslash\mathrm{tl_set_eq:NN}$	1474	$\um@usv@bfitvarphi$ 104, 531, 539, 1084, 1121, 1146
$\backslash\mathrm{tl_use:c}$	894	$\um@usv@bfitvarpi$ 106, 533, 541, 1086, 1123, 1148
$\backslash\mathrm{to}$	1517	$\um@usv@bfitvarrho$ 105, 532, 540, 1085, 1122, 1147
$\backslash\mathrm{token_if_eq_catcode_p:NN}$..	1484, 1485	$\um@usv@bfitvarTheta$ 100, 521, 524, 1078, 1105, 1140
$\backslash\mathrm{token_if_eq_meaning_p:NN}$	1486	$\um@usv@bfitvartheta$ 102, 529, 537, 1082, 1119, 1144
$\backslash\mathrm{ttdefault}$	919	$\um@usv@bfLatin$	50
U			
$\um@addto@mathmap$	549, 556, <u>561</u>	$\um@usv@bfNabla$ 109, 212, 461, 470, 1068, 1161
$\um@backslash$	629, 652	$\um@usv@bfnum$ 48, 1054, 1131, 1151, 1171, 1176
$\um@char@num@range$ 372, 553, 641, 642, 644		$\um@usv@bfpartial$	115, 230, 465, 472, 1070, 1115, 1162
$\um@char@range$ 371, 409, 496, 619, 622, 625		$\um@usv@bfscrLatin$	60, 1177
$\um@firstchar$	628, 653	$\um@usv@bfscrLatin$	61, 1178
$\um@firsttof$	651–653	$\um@usv@bfsfitGreek$ 70, 1189, 1229, 1262	
$\um@font$	311, 314, 399, 400, 896, 912, 925, 1360, 1362, 1363, 1366	$\um@usv@bfsfitgreek$ 71, 1190, 1230, 1263	
$\um@fontdimen@percent$ <u>306</u> , 311, 314, 1362, 1363	$\um@usv@bfsfitLatin$ 68, 1185, 1227, 1260	
$\um@mathsymbol$	<u>322</u> , 438	$\um@usv@bfsfitlatin$ 69, 1186, 1228, 1261	
$\um@mversion$	375, 376	$\um@usv@bfsfitNabla$	112, 217, 464, 471, 1201, 1232, 1265
$\um@nolimits$	336, <u>567</u> , 575, 583	$\um@usv@bfsfitpartial$	118, 235, 468, 473, 1202, 1233, 1266
$\um@parse@range$	638, <u>654</u>	$\um@usv@bfsfLatin$	64
$\um@parse@term$	441, 477, <u>624</u>		
$\um@radicals$	342, <u>586</u>		
$\um@resolve@greek$	<u>728</u>		
$\um@scaled@apply$	<u>309</u> , 1360, 1366		
$\um@scanactivedef$	329, <u>1274</u>		
$\um@scancharlet$	<u>1274</u> , 1294		
$\um@set@mathsymbol$	323, <u>324</u>		
$\um@usv@bbLatin$	33, 981		
$\um@usv@bbLatin$	34, 989		
$\um@usv@bbnum$	32, 980		
$\um@usv@bfDigamma$	82, 1069		
$\um@usv@bfDigamma$	89, 1077		
$\um@usv@bfrakLatin$	58, 1172		

$\backslash\mathrm{um@usv@bfsfNabla}$	111, 213, 463, 471	1089, 1091, 1132, 1152, 1172, 1177, 1185, 1212, 1227, 1244, 1260
$\backslash\mathrm{um@usv@bfsfnum}$		
. . .	62, 1182, 1211, 1226, 1243, 1259	
$\backslash\mathrm{um@usv@bfsfpartial}$. .	117, 231, 467, 473	$\backslash\mathrm{um@usv@itlatin}$ 29, 798, 825, 828, 946, 965, 989, 1003, 1017, 1025, 1030, 1034, 1041, 1046, 1051, 1061, 1094, 1097, 1133, 1153, 1173, 1178, 1186, 1213, 1228, 1245, 1261
$\backslash\mathrm{um@usv@bfsfupGreek}$ 66,	1187, 1214, 1246	
$\backslash\mathrm{um@usv@bfsfupgreek}$ 67,	1188, 1215, 1247	
$\backslash\mathrm{um@usv@bfsfupLatin}$ 63,	1183, 1212, 1244	$\backslash\mathrm{um@usv@itNabla}$ 108, 215, 453, 457, 950, 969, 1079, 1126, 1141, 1161, 1201, 1217, 1232, 1249, 1265
$\backslash\mathrm{um@usv@bfsfuplatin}$ 65,	1184, 1213, 1245	
$\backslash\mathrm{um@usv@bfupGreek}$	52, 520, 523, 811, 1062, 1101, 1138, 1154, 1158	$\backslash\mathrm{um@usv@itpartial}$ 114, 233, 455, 458, 955, 970, 1080, 1115, 1124, 1127, 1142, 1162, 1202, 1218, 1233, 1250, 1266
$\backslash\mathrm{um@usv@bfupgreek}$	53, 527, 535, 812, 1064, 1108, 1139, 1155, 1159	$\backslash\mathrm{um@usv@itvarepsilon}$. 92, 844, 852, 956, 972, 1081, 1109, 1118, 1143, 1163, 1203, 1219, 1234, 1251, 1267
$\backslash\mathrm{um@usv@bfuph}$	98, 1095	$\backslash\mathrm{um@usv@itvarkappa}$. . . 94, 846, 854, 958, 974, 1083, 1111, 1120, 1145, 1165, 1205, 1221, 1236, 1253, 1269
$\backslash\mathrm{um@usv@bfupLatin}$	49, 510, 512, 807, 1058, 1089, 1136, 1152, 1156	$\backslash\mathrm{um@usv@itvarphi}$ 95, 847, 855, 959, 975, 1084, 1112, 1121, 1146, 1166, 1206, 1222, 1237, 1254, 1270
$\backslash\mathrm{um@usv@bfuplatin}$	51, 515, 517, 808, 1060, 1094, 1137, 1153, 1157	$\backslash\mathrm{um@usv@itvarpi}$ 97, 849, 857, 961, 977, 1086, 1114, 1123, 1148, 1168, 1208, 1224, 1239, 1256, 1272
$\backslash\mathrm{um@usv@bfvarepsilon}$		$\backslash\mathrm{um@usv@itvarrho}$ 96, 848, 856, 960, 976, 1085, 1113, 1122, 1147, 1167, 1207, 1223, 1238, 1255, 1271
.	83, 528, 536, 1071, 1109, 1163	$\backslash\mathrm{um@usv@itvarTheta}$ 91, 838, 951, 971, 1078, 1102, 1105, 1140, 1160, 1216, 1248
$\backslash\mathrm{um@usv@bfvarkappa}$		$\backslash\mathrm{um@usv@itvartheta}$. . . 93, 845, 853, 957, 973, 1082, 1110, 1119, 1144, 1164, 1204, 1220, 1235, 1252, 1268
.	85, 530, 538, 1073, 1111, 1165	$\backslash\mathrm{um@usv@Nabla}$ 107, 211, 452, 457, 950, 969, 1068, 1126, 1141, 1161, 1192, 1217, 1232, 1249, 1265
$\backslash\mathrm{um@usv@bfvarphi}$		$\backslash\mathrm{um@usv@num}$ 23, 497, 980, 1021, 1028, 1032, 1039, 1044, 1049, 1054, 1131, 1151, 1171, 1176, 1182, 1211, 1226, 1243, 1259
.	86, 531, 539, 1074, 1112, 1166	$\backslash\mathrm{um@usv@partial}$ 113, 229, 454, 458, 955, 970, 1070, 1115, 1124, 1127, 1142, 1162, 1193, 1218, 1233, 1250, 1266
$\backslash\mathrm{um@usv@bfvarpi}$		$\backslash\mathrm{um@usv@scrLatin}$ 35, 992
.	88, 533, 541, 1076, 1114, 1168	
$\backslash\mathrm{um@usv@bfvarrho}$		
.	87, 532, 540, 1075, 1113, 1167	
$\backslash\mathrm{um@usv@bfvarTheta}$		
.	81, 521, 524, 1067, 1102, 1160	
$\backslash\mathrm{um@usv@bfvartheta}$		
.	84, 529, 537, 1072, 1110, 1164	
$\backslash\mathrm{um@usv@Digamma}$	73, 1055, 1069	
$\backslash\mathrm{um@usv@digamma}$	80, 1056, 1077	
$\backslash\mathrm{um@usv@frakLatin}$	37, 1009, 1172	
$\backslash\mathrm{um@usv@fraklatin}$	38, 1017, 1173	
$\backslash\mathrm{um@usv@itGreek}$. . 30,	803, 834, 837, 949, 967, 1063, 1101, 1104, 1134, 1154, 1189, 1214, 1229, 1246, 1262	
$\backslash\mathrm{um@usv@itgreek}$	31, 843, 851, 954, 968, 1065, 1108, 1117, 1135, 1155, 1190, 1215, 1230, 1247, 1263	
$\backslash\mathrm{um@usv@ith}$	90, 799, 826, 829, 966, 1066, 1095, 1098	
$\backslash\mathrm{um@usv@itLatin}$ 28,	797, 818, 820, 943, 964, 981, 992, 1009, 1024, 1029, 1033, 1040, 1045, 1050, 1059,	

<code>\um@usv@scrLatin</code>	36, 1003	957, 973, 1072, 1110, 1119, 1144,
<code>\um@usv@sfItLatin</code> ..	43, 1024, 1033, 1045	1164, 1195, 1220, 1235, 1252, 1268
<code>\um@usv@sfItLatin</code> ..	44, 1025, 1034, 1046	<code>\um@zf@feature</code>
<code>\um@usv@sfLatin</code>	41	595, 607, 610
<code>\um@usv@sfnum</code>		<code>\um_bfNabla_up_or_it_usv</code>
...	39, 1021, 1028, 1032, 1039, 1044	212, 216, 470, 1126
<code>\um@usv@sfupLatin</code> ..	40, 1022, 1029, 1040	<code>\um_bfpartial_up_or_it_usv</code>
<code>\um@usv@sfupLatin</code> ..	42, 1023, 1030, 1041	230, 234, 472, 1127
<code>\um@usv@ttLatin</code>	46, 1050	<code>\um_bfsfNabla_up_or_it_usv</code> 213, 217, 471
<code>\um@usv@ttLatin</code>	47, 1051	<code>\um_bfsfpartial_up_or_it_usv</code>
<code>\um@usv@ttnum</code>	45, 1049	231, 235, 473
<code>\um@usv@upGreek</code> ..	26, 801, 834, 837,	<code>\um_config_mathbb:</code>
949, 967, 1062, 1101, 1104, 1134,		979
1154, 1187, 1214, 1229, 1246, 1262		<code>\um_config_mathbf:</code>
<code>\um@usv@upgreek</code> ..	27, 804, 843, 851,	1053
954, 968, 1064, 1108, 1117, 1135,		<code>\um_config_mathbffrak:</code>
1155, 1188, 1215, 1230, 1247, 1263		1170
<code>\um@usv@upLatin</code> 24, 796, 818, 820, 943,		<code>\um_config_mathbfit:</code>
964, 981, 992, 1009, 1022, 1029,		1130
1033, 1040, 1045, 1050, 1058,		<code>\um_config_mathbfscr:</code>
1089, 1091, 1132, 1152, 1172,		1175
1177, 1183, 1212, 1227, 1244, 1260		<code>\um_config_mathbfsf:</code>
<code>\um@usv@upLatin</code> 25, 800, 825, 828, 946,		1180
965, 989, 1003, 1017, 1023, 1030,		<code>\um_config_mathbfsfit:</code>
1034, 1041, 1046, 1051, 1060,		1258
1094, 1097, 1133, 1153, 1173,		<code>\um_config_mathbfsfup:</code>
1178, 1184, 1213, 1228, 1245, 1261		1242
<code>\um@usv@varepsilon</code> ...	74, 844, 852,	<code>\um_config_mathbfup:</code>
956, 972, 1071, 1109, 1118, 1143,		1150
1163, 1194, 1219, 1234, 1251, 1267		<code>\um_config_mathfrak_Latin:</code>
<code>\um@usv@varkappa</code>	76, 846, 854,	1008
958, 974, 1073, 1111, 1120, 1145,		<code>\um_config_mathfrak_Latin:</code>
1165, 1196, 1221, 1236, 1253, 1269		1016
<code>\um@usv@varphi</code>	77, 847, 855,	<code>\um_config_mathit:</code>
959, 975, 1074, 1112, 1121, 1146,		963
1166, 1197, 1222, 1237, 1254, 1270		<code>\um_config_mathscr_Latin:</code>
<code>\um@usv@varpi</code>	79, 849, 857,	991
961, 977, 1076, 1114, 1123, 1148,		<code>\um_config_mathscr_Latin:</code>
1168, 1199, 1224, 1239, 1256, 1272		1002
<code>\um@usv@varrho</code>	78, 848, 856,	<code>\um_config_mathsf:</code>
960, 976, 1075, 1113, 1122, 1147,		1019
1167, 1198, 1223, 1238, 1255, 1271		<code>\um_config_mathsf:</code>
<code>\um@usv@varTheta</code> 72, 802, 835, 838, 951,		1043
971, 1067, 1102, 1105, 1140, 1160,		<code>\um_config_mathsfup:</code>
1191, 1200, 1216, 1231, 1248, 1264		1038
<code>\um@usv@vartheta</code>	75, 845, 853,	<code>\um_config_mathhtt:</code>
		1048
		<code>\um_config_mathup_Greek:</code>
		948
		<code>\um_config_mathup_Greek:</code>
		953
		<code>\um_config_mathup_Latin:</code>
		942
		<code>\um_config_mathup_Latin:</code>
		945
		<code>\um_fix_mathtt:</code>
		918
		<code>\um_glyph_if_exist:n</code>
		924
		<code>\um_glyph_if_exist:nT</code>
		881
		<code>\um_glyph_if_exist:nTF</code>
	 888, 906, <u>924</u> , 1318, 1321, 1324
		<code>\um_init_alphabet:n</code>
		415, 921
		<code>\um_make_mathactive:nNN</code>
		487, <u>489</u>
		<code>\um_map_char:nn</code>
		521, 524,
		528–533, 536–541, 700, 799, 802,
		826, 829, 835, 838, 844–849, 852–857
		<code>\um_map_char:nn_</code>
		<u>681</u>
		<code>\um_map_chars_greek:nn</code>
		... 520, 523, 527, 535, 694, 801,
		803, 804, 811–814, 834, 837, 843, 851

<code>\um_map_chars_latin:nn</code>	1191–1208, 1216–1224, 1231–1239, 1248–1256, 1264–1272
510, 512, 515, 517, 691, 796–798, 800, 807–810, 818, 820, 825, 828	<code>\um_set_mathalphabet_char:Nnnn</code> .. <u>703</u>
<code>\um_map_chars_numbers:nn</code> 497, 697	<code>\um_set_mathalphabet_greek:Nnn</code> ..
<code>\um_map_chars_range:nnn</code> 725, 949,
..... 681, 692, 695, 698, 701	954, 967, 968, 1062–1065, 1101,
<code>\um_mathmap:Nnn</code> 413, 420, 706, 713	1104, 1108, 1117, 1134, 1135,
<code>\um_mathmap_noparse:Nnn</code> 413, <u>547</u>	1138, 1139, 1154, 1155, 1158,
<code>\um_mathmap_parse:Nnn</code> 420, <u>552</u>	1159, 1187–1190, 1214, 1215,
<code>\um_maybe_init_alphabet:n</code>	1229, 1230, 1246, 1247, 1262, 1263
..... 415, 422, 882, 907	<code>\um_set_mathalphabet_latin:Nnn</code> ..
<code>\um_Nabla_up_or_it_usv</code> .. 211, 215, 457 722, 943, 946, 964,
<code>\um_nprimes:n</code> 1308, 1318, 1321, 1324, 1327	965, 981, 989, 992, 1003, 1009,
<code>\um_nprimes_select:n</code> 1314, 1345	1017, 1022–1025, 1029, 1030,
<code>\um_partial_up_or_it_usv</code> . 229, 233, 458	1033, 1034, 1040, 1041, 1045,
<code>\um_peek_execute_branches:ss</code> ...	1046, 1050, 1051, 1058–1061,
..... 1477, 1482	1089, 1091, 1094, 1097, 1132,
<code>\um_peek_execute_branches_ss_aux:</code>	1133, 1136, 1137, 1152, 1153,
..... 1489, 1502	1156, 1157, 1172, 1173, 1177,
<code>\um_prepare_alph:n</code> 883, 908, <u>927</u>	1178, 1183–1186, 1212, 1213,
<code>\um_process_symbol_noparse:nnnn</code> ..	1227, 1228, 1244, 1245, 1260, 1261
..... 412, <u>437</u>	<code>\um_set_mathalphabet_numbers:Nnn</code> .
<code>\um_process_symbol_parse:nnnn</code> 419, <u>437</u> 719,
<code>\um_remap_symbol:nnn</code>	980, 1021, 1028, 1032, 1039, 1044,
..... 414, 421, 446, 447, 449,	1049, 1054, 1131, 1151, 1171,
452–455, 457, 458, 461–468, 470–473	1176, 1182, 1211, 1226, 1243, 1259
<code>\um_remap_symbol_noparse:nnn</code> 414, <u>445</u>	<code>\um_set_mathcode:nnnn</code> <u>366</u> , 483, 564, 684
<code>\um_remap_symbol_parse:nnn</code> 421, <u>445</u> , 476	<code>\um_setup_active_subscript:nn</code> ...
<code>\um_remap_symbols:</code> 428, <u>445</u> 1416, 1430–1457
<code>\um_scan_sscript:</code> 1392, 1425, 1462, 1464	<code>\um_setup_active_superscript:nn</code> ..
<code>\um_scan_sscript:TF</code> 1463, 1472 1383, 1397–1413
<code>\um_scanprime:</code>	<code>\um_setup_alphabets:</code> 431, 860
..... 1330, 1339, 1350, 1354, 1355	<code>\um_setup_alphanum:</code> 430, <u>495</u>
<code>\um_scanprime_collect:</code>	<code>\um_setup_bf_literals:</code> 507, <u>806</u>
..... 1332, 1334, 1337, 1340, 1343	<code>\um_setup_Greek:</code> 503, <u>832</u>
<code>\um_set_mathalph_range:Nnn</code> <u>710</u>	<code>\um_setup_greek:</code> 504, <u>841</u>
<code>\um_set_mathalph_range:nNnn</code>	<code>\um_setup_Latin:</code> 501, <u>816</u>
..... 710, 720, 723, 726	<code>\um_setup_latin:</code> 502, <u>823</u>
<code>\um_set_mathalphabet_char:Nnn</code> ...	<code>\um_setup_literals:</code> 499, <u>795</u>
..... 704, 950, 951,	<code>\um_setup_math_alphabet:n</code>
955–961, 966, 969–977, 982–988, 862, 863, 866–877, 905
993–1000, 1004–1006, 1010–1014,	<code>\um_setup_math_alphabet:nn</code>
1055, 1056, 1066–1086, 861, 864, 865, <u>879</u>
1095, 1098, 1102, 1105,	<code>\um_setup_mathactives:</code> 429, <u>486</u>
1109–1115, 1118–1124, 1126,	<code>\um_setup_mathup:</code> 1516
1127, 1140–1148, 1160–1168,	<code>\um_setup_nabla:</code> 209, 434

