

Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/10/02 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’	3			
3.2	Bold style	4			
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	12			
4.1	Package options	16			
4.2	Overcoming \@on-lypreamble	20			
			4.3	Other things	21
5	Fundamentals	22			
5.1	Enlarging the number of maths families	22			
5.2	\DeclareMathSymbol for unicode ranges	22			
5.3	The main \setmathfont macro	24			
5.4	(Big) operators	32			
5.5	Radicals	36			
5.6	Delimiters	36			
5.7	Maths accents	38			
6	Font features	39			
6.1	OpenType maths font features	40			
6.2	Script and scriptscript font options	40			
6.3	Range processing	40			
6.4	Resolving Greek symbol name control sequences	44			

6.5	Setting up the mappings	46	II	stix table data extraction	69
7	Maths alphabets mapping definitions	48	A	Documenting maths support in the NFSS	71
7.1	Non-bold math alphabets	50	A.1	Overview	71
7.2	Bold math alphabets	54			
7.3	Definitions of the math symbols	60	III	X_YTeX math font dimensions	72
8	Epilogue	61			

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 $\mathit{\xi}$, \leq to $\mathit{\leq}$, 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

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

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

2.1 Using multiple fonts

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

`\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 `\J`, 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.

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

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

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 L^AT_EX’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.

3.2 Bold style

Similar as in the previous section, ISO standards differ somewhat to T_EX’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

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>	$(\mathbf{a}, \mathbf{z}, \mathbf{B}, \mathbf{X})$	$(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\Gamma}, \boldsymbol{\Xi})$

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{bf}` in the former (`'M'`), and `\bm` (or `\boldsymbol`, deprecated) in the latter (`'ξ'`).

In unicode-math, the `\mathbf{bf}` 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`, `\mathbf{sfup}`, and `\mathbf{sfit}` 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 \LaTeX '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 `\mathsfif`, 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 `\mathbfsfit` 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). \TeX classically uses an upright nabla, but iso standards differ (I think). The package options `nabla=upright` and `nabla=italic` switch between the two choices. This is then inherited through `\mathbf`; `\mathit` and `\mathup` can be used to force one way or the other.

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

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>	•		

Table 4: The various forms of nabla.

Description		Glyph
Upright	Serif	∇
	Bold serif	▮
	Bold sans	∇
Italic	Serif	∇
	Bold serif	▮
	Bold sans	∇

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	∂
Bold	Upright	∂
	Italic	∂
Sans bold	Upright	∂
	Italic	∂

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.

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

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

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

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

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.

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.

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.

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: ‘ $a:b$ ’.

$\text{A}_0 \text{ }_1 \text{ }_2 \text{ }_3 \text{ }_4 \text{ }_5 \text{ }_6 \text{ }_7 \text{ }_8 \text{ }_9 \text{ }_{+} \text{ }_{-} \text{ }_{=} \text{ }_{(} \text{ }_{)} \text{ }_a \text{ }_e \text{ }_i \text{ }_o \text{ }_r \text{ }_u \text{ }_v \text{ }_x \text{ }_\beta \text{ }_\gamma \text{ }_\rho \text{ }_\varphi \text{ }_\chi \text{Z}$

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

Table 6: Slashes and backslashes.

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

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. The complete list is shown in table 6.

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).

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.

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

U+29F8: BIG SOLIDUS is a ‘big operator’ (like Σ).

Backslash The U+005C: REVERSE SOLIDUS character `\backslash` is used for denoting double cosets: $A \backslash B$. (So I’m led to believe.) It may be used as a ‘stretchy’ delimiter if supported by the font.

MathML uses U+2216: SET MINUS like this: $A \setminus B$.² The L^AT_EX command name `\smallsetminus` is used for backwards compatibility.

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 The L^AT_EX name for this character is `\setminus`.

Finally, U+29F9: BIG REVERSE SOLIDUS is a ‘big operator’ (like Σ).

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

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

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

Of the above characters, the following are allowed to be used after `\left`, `\middle`, and `\right`:

- `\solidus`;
- `\fracslash`;
- `\slash`; and,
- `\backslash` (the only reverse slash).

However, we assume that there is only *one* stretchy slash in the font; this is assumed by default to be U+002F: SOLIDUS. Writing `\left/` or `\left\slash` or `\leftfracslash` will all result in the same stretchy delimiter being used.

The delimiter used can be changed with the `slash-delimiter` package option. Allowed values are `ascii`, `frac`, and `div`, corresponding to the respective unicode slots.

²§4.4.5.11 [https://www.w3.org/TR/2003/REC-math3/](#)

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

For example: as mentioned above, Cambria Math’s stretchy slash is U+2044: FRACTION SLASH. When using Cambria Math, then unicode-math should be loaded with the [slash-delimiter=frac] option. (This should be a font option rather than a package option, but it will change soon.)

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 [2009/10/02 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 L^AT_EX3 — 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}
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@sfnun{"1D7E2}
40 \def\um@usv@sfnunum{"1D7E2}
```

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

```

41 \def\um@usv@sfitnum{"1D7E2}
42 \def\um@usv@sfupLatin{"1D5A0}
43 \def\um@usv@sflatin {"1D5A0}
44 \def\um@usv@sfuplatin{"1D5BA}
45 \def\um@usv@sflatin{"1D5BA}
46 \def\um@usv@sfitLatin{"1D608}
47 \def\um@usv@sfitlatin{"1D622}
48 \def\um@usv@ttnum{"1D7F6}
49 \def\um@usv@ttLatin{"1D670}
50 \def\um@usv@ttlatin{"1D68A}

```

Bold:

```

51 \def\um@usv@bfnum {"1D7CE}
52 \def\um@usv@bfupnum{"1D7CE}
53 \def\um@usv@bfitnum{"1D7CE}
54 \def\um@usv@bfupLatin{"1D400}
55 \def\um@usv@bfLatin {"1D400}
56 \def\um@usv@bfuplatin{"1D41A}
57 \def\um@usv@bflatin {"1D41A}
58 \def\um@usv@bfupGreek{"1D6A8}
59 \def\um@usv@bfupgreek{"1D6C2}
60 \def\um@usv@bfGreek {"1D6A8}
61 \def\um@usv@bfgreek {"1D6C2}
62 \def\um@usv@bfitLatin{"1D468}
63 \def\um@usv@bfitlatin{"1D482}
64 \def\um@usv@bfitGreek{"1D71C}
65 \def\um@usv@bfitgreek{"1D736}
66 \def\um@usv@bffrakLatin{"1D56C}
67 \def\um@usv@bffraklatin{"1D586}
68 \def\um@usv@bfscrLatin{"1D4D0}
69 \def\um@usv@bfscrlatin{"1D4EA}
70 \def\um@usv@bfsfnum {"1D7EC}
71 \def\um@usv@bfsfupnum{"1D7EC}
72 \def\um@usv@bfsfitnum{"1D7EC}
73 \def\um@usv@bfsfupLatin{"1D5D4}
74 \def\um@usv@bfsfLatin {"1D5D4}
75 \def\um@usv@bfsfuplatin{"1D5EE}
76 \def\um@usv@bfsflatin {"1D5EE}
77 \def\um@usv@bfsfupGreek{"1D756}
78 \def\um@usv@bfsfupgreek{"1D770}
79 \def\um@usv@bfsfGreek {"1D756}
80 \def\um@usv@bfsfgreek {"1D770}
81 \def\um@usv@bfsfitLatin{"1D63C}
82 \def\um@usv@bfsfitlatin{"1D656}
83 \def\um@usv@bfsfitGreek{"1D790}
84 \def\um@usv@bfsfitgreek{"1D7AA}

```

Greek variants:

```

85 \def\um@usv@varTheta{"3F4}
86 \def\um@usv@Digamma{"3DC}
87 \def\um@usv@varepsilon{"3F5}
88 \def\um@usv@vartheta{"3D1}
89 \def\um@usv@varkappa{"3F0}
90 \def\um@usv@varphi{"3D5}
91 \def\um@usv@varrho{"3F1}
92 \def\um@usv@varpi{"3D6}
93 \def\um@usv@digamma{"3DD}

```

Bold:

```

94 \def\um@usv@bfvarTheta{"1D6B9}
95 \def\um@usv@bfDigamma{"1D7CA}
96 \def\um@usv@bfvarepsilon{"1D6DC}
97 \def\um@usv@bfvartheta{"1D6DD}
98 \def\um@usv@bfvarkappa{"1D6DE}
99 \def\um@usv@bfvarphi{"1D6DF}
100 \def\um@usv@bfvarrho{"1D6E0}
101 \def\um@usv@bfvarpi{"1D6E1}
102 \def\um@usv@bfdigamma{"1D7CB}

```

Italic Greek variants:

```

103 \def\um@usv@ith{"210E}
104 \def\um@usv@itvarTheta{"1D6F3}
105 \def\um@usv@itvarepsilon{"1D716}
106 \def\um@usv@itvartheta{"1D717}
107 \def\um@usv@itvarkappa{"1D718}
108 \def\um@usv@itvarphi{"1D719}
109 \def\um@usv@itvarrho{"1D71A}
110 \def\um@usv@itvarpi{"1D71B}

```

Bold italic:

```

111 \def\um@usv@bfuph{"1D421}
112 \def\um@usv@bfith{"1D489}
113 \def\um@usv@bfitvarTheta{"1D72D}
114 \def\um@usv@bfitvarepsilon{"1D750}
115 \def\um@usv@bfitvartheta{"1D751}
116 \def\um@usv@bfitvarkappa{"1D752}
117 \def\um@usv@bfitvarphi{"1D753}
118 \def\um@usv@bfitvarrho{"1D754}
119 \def\um@usv@bfitvarpi{"1D755}

```

Nabla:

```

120 \def\um@usv@Nabla{"2207}
121 \def\um@usv@itNabla{"1D6FB}
122 \def\um@usv@bfNabla{"1D6C1}
123 \def\um@usv@bfitNabla{"1D735}
124 \def\um@usv@bfsfNabla{"1D76F}
125 \def\um@usv@bfsfitNabla{"1D7A9}

```

Partial:

```
126 \def\um@usv@partial{"2202}
127 \def\um@usv@itpartial{"1D715}
128 \def\um@usv@bfpartial{"1D6DB}
129 \def\um@usv@bfitpartial{"1D74F}
130 \def\um@usv@bfsfpartial{"1D789}
131 \def\um@usv@bfsfitpartial{"1D7C3}
```

4.1 Package options

xkeyval's package support is used here.

math-style

```
132 \define@choicekey*{unicode-math.sty}
133   {math-style}[\@tempa\@tempb]{iso,tex,french,literal}{
134   \ifcase\@tempb\relax
135     \@um@upGreekfalse
136     \@um@upgreekfalse
137     \@um@upLatinfalse
138     \@um@uplatinfalse
139     \@um@bfupGreekfalse
140     \@um@bfupgreekfalse
141     \@um@uppartialfalse
142     \@um@bfupLatinfalse
143     \@um@bfuplatinfalse
144     \@um@upNablafalse
145     \bool_set_false:N \g_um_upsans_bool
146     \bool_set_false:N \g_um_texgreek_bool
147   \or
148     \@um@upGreektrue
149     \@um@upgreekfalse
150     \@um@upLatinfalse
151     \@um@uplatinfalse
152     \@um@bfupGreektrue
153     \@um@bfupgreekfalse
154     \@um@uppartialfalse
155     \@um@bfupLatintrue
156     \@um@bfuplatintrue
157     \@um@upNablatrue
158     \bool_set_true:N \g_um_upsans_bool
159     \bool_set_true:N \g_um_texgreek_bool
160   \or
161     \@um@upGreektrue
162     \@um@upgreektrue
163     \@um@upLatintrue
```



```

164 \um@uplatinfalse
165 \um@bfupGreektrue
166 \um@bfupgreektrue
167 \um@uppartialtrue
168 \um@bfupLatintrue
169 \um@bfuplatintrue
170 \um@upNablatrue
171 \bool_set_true:N \g_um_upsans_bool
172 \bool_set_false:N \g_um_texgreek_bool
173 \or
174 \um@literaltrue
175 \um@bfliteraltrue
176 \bool_set_true:N \g_um_sfliteral_bool
177 \bool_set_false:N \g_um_texgreek_bool
178 \fi
179 }

```

bold-style

```

180 \define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,french,literal}{
181 \ifcase\@tempb\relax
182 \um@bfupGreekfalse
183 \um@bfupgreekfalse
184 \um@bfupLatinfalse
185 \um@bfuplatinfalse
186 \um@uppartialfalse
187 \or
188 \um@bfupGreektrue
189 \um@bfupgreekfalse
190 \um@bfupLatintrue
191 \um@bfuplatintrue
192 \um@uppartialfalse
193 \or
194 \um@bfupGreektrue
195 \um@bfupgreektrue
196 \um@bfupLatintrue
197 \um@bfuplatintrue
198 \um@uppartialtrue
199 \or
200 \um@bfliteraltrue
201 \fi
202 }
203 \cs_set:Nn \um_setup_bfshapes: {
204 \tl_set:Nx \um_bf_Greek_up_or_it_usv { \ifum@bfupGreek \um@usv@bfupGreek \else \um@usv@bfit
205 \tl_set:Nx \um_bf_greek_up_or_it_usv { \ifum@bfupgreek \um@usv@bfupgreek \else \um@usv@bfit
206 \tl_set:Nx \um_bf_Latin_up_or_it_usv { \ifum@bfupLatin \um@usv@bfupLatin \else \um@usv@bfit
207 \tl_set:Nx \um_bf_latin_up_or_it_usv { \ifum@bfuplatin \um@usv@bfuplatin \else \um@usv@bfit

```

208 }

sans-style

```

209 \bool_new:N \g_um_upsans_bool
210 \bool_new:N \g_um_sfliteral_bool
211 \define@choicekey*{unicode-math.sty}
212   {sans-style}[\@tempa\@tempb]{iso,tex,literal}{
213   \ifcase\@tempb\relax
214     \bool_set_false:N \g_um_upsans_bool
215   \or
216     \bool_set_true:N \g_um_upsans_bool
217   \or
218     \bool_set_true:N \g_um_sfliteral_bool
219   \fi
220 }
221 \cs_set:Nn \um_setup_sfshapes: {
222   \bool_if:NTF \g_um_upsans_bool {
223     \tl_set:Nn \um_sf_Latin_up_or_it_usv      { \um@usv@sflatin    }
224     \tl_set:Nn \um_sf_latin_up_or_it_usv      { \um@usv@sflatin    }
225     \tl_set:Nn \um_bfsf_Latin_up_or_it_usv    { \um@usv@bfsfupLatin  }
226     \tl_set:Nn \um_bfsf_latin_up_or_it_usv    { \um@usv@bfsfupLatin  }
227     \tl_set:Nn \um_bfsf_Greek_up_or_it_usv    { \um@usv@bfsfupGreek  }
228     \tl_set:Nn \um_bfsf_greek_up_or_it_usv    { \um@usv@bfsfupgreek  }
229   }{
230     \tl_set:Nn \um_sf_Latin_up_or_it_usv      { \um@usv@sfitLatin    }
231     \tl_set:Nn \um_sf_latin_up_or_it_usv      { \um@usv@sfitlatin    }
232     \tl_set:Nn \um_bfsf_Latin_up_or_it_usv    { \um@usv@bfsffitLatin  }
233     \tl_set:Nn \um_bfsf_latin_up_or_it_usv    { \um@usv@bfsffitLatin  }
234     \tl_set:Nn \um_bfsf_Greek_up_or_it_usv    { \um@usv@bfsfitGreek  }
235     \tl_set:Nn \um_bfsf_greek_up_or_it_usv    { \um@usv@bfsfitgreek  }
236   }
237 }

```

Symbol obliqueness

```

238 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
239   \ifcase\@tempb\relax
240     \@um@upNablatrue
241   \or
242     \@um@upNablafalse
243   \fi
244 }
245 \cs_set:Nn \um_setup_nabla: {
246   \if@um@upNabla
247     \tl_set:Nn \um_Nabla_up_or_it_usv      { \um@usv@Nabla  }
248     \tl_set:Nn \um_bfNabla_up_or_it_usv    { \um@usv@bfNabla  }

```

```

249 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfNabla }
250 \else
251 \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@itNabla }
252 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfitNabla }
253 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfitNabla }
254 \fi
255 }

256 \define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
257 \ifcase\@tempb\relax
258 \um@uppartialtrue
259 \or
260 \um@uppartialfalse
261 \fi
262 }
263 \cs_set:Nn \um_setup_partial: {
264 \if@um@uppartial
265 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@partial }
266 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfpartial }
267 \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfpartial }
268 \else
269 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@itpartial }
270 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfitpartial }
271 \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfitpartial }
272 \fi
273 }

```

Epsilon and phi shapes

```

274 \define@choicekey*{unicode-math.sty}{vargreek-shape}[\@tempa\@tempb]{unicode,TeX}{
275 \ifcase\@tempb\relax
276 \bool_set_false:N \g_um_texgreek_bool
277 \or
278 \bool_set_true:N \g_um_texgreek_bool
279 \fi
280 }

```

Colon style

```

281 \bool_new:N \g_um_literal_colon_bool
282 \define@choicekey*{unicode-math.sty}{colon}[\@tempa\@tempb]{literal,TeX}{
283 \ifcase\@tempb\relax
284 \bool_set_true:N \g_um_literal_colon_bool
285 \or
286 \bool_set_false:N \g_um_literal_colon_bool
287 \fi
288 }

```

Slash delimiter style

```
289 \define@choicekey*{unicode-math.sty}{slash-delimiter}[\@tempa\@tempb]{ascii,frac,div}{
290   \ifcase\@tempb\relax
291     \tl_set:Nn \g_um_slash_delimiter_usv {"002F}
292   \or
293     \tl_set:Nn \g_um_slash_delimiter_usv {"2044}
294   \or
295     \tl_set:Nn \g_um_slash_delimiter_usv {"2215}
296   \fi
297 }

298 \ExecuteOptionsX{math-style=TeX,slash-delimiter=ascii}
299 \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.

```
300 \tl_map_inline:nn {
301   \new@mathgroup
302   \cdp@list
303   \cdp@elt
304   \DeclareMathSizes
305   \@DeclareMathSizes
306   \newmathalphabet
307   \newmathalphabet@@
308   \newmathalphabet@@@
309   \DeclareMathVersion
310   \define@mathalphabet
311   \define@mathgroup
312   \addtoversion
313   \version@list
314   \version@elt
315   \alpha@list
316   \alpha@elt
317   \restore@mathversion
318   \init@restore@version
319   \dorestore@version
320   \process@table
321   \new@mathversion
322   \DeclareSymbolFont
323   \group@list
324   \group@elt
325   \new@symbolfont
326   \SetSymbolFont
327   \SetSymbolFont@
```

```

328 \get@cdp
329 \DeclareMathAlphabet
330 \new@mathalphabet
331 \SetMathAlphabet
332 \SetMathAlphabet@
333 \DeclareMathAccent
334 \set@mathaccent
335 \DeclareMathSymbol
336 \set@mathchar
337 \set@mathsymbol
338 \DeclareMathDelimiter
339 \@xxDeclareMathDelimiter
340 \@DeclareMathDelimiter
341 \@xDeclareMathDelimiter
342 \set@mathdelimiter
343 \set@@mathdelimiter
344 \DeclareMathRadical
345 \mathchar@type
346 \DeclareSymbolFontAlphabet
347 \DeclareSymbolFontAlphabet@
348 }{
349 \tl_remove_in:Nn \@preamblecmds {\do#1}
350 }

```

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

```

351 \def\um@fontdimen@percent#1#2{
352   0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax
353 }

```

\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.

```

354 \def\um@scaled@apply#1#2#3{

```

```

355 \ifx#1\scriptstyle
356   #2\um@fontdimen@percent{10}\um@font#3
357 \else
358   \ifx#1\scriptscriptstyle
359     #2\um@fontdimen@percent{11}\um@font#3
360   \else
361     #2#3%
362   \fi
363 \fi
364 }

```

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

```

365 \def\new@mathgroup{\alloc@8\mathgroup\chardef\@cclvi}
366 \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
367 \def \um@mathsymbol#1#2#3#4{
368   \expandafter\um@set@mathsymbol\csname sym#3\endcsname#1#2{#4}}

```

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

```

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

```

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

```
370 \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`.

```
371 \begingroup
372 \char_make_active:n {#4}
373 \global\mathcode#4="8000\relax
374 \um@scanactivedef #4 \@nil { \csname\cs_to_str:N #2 op\endcsname }
375 \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`.

```
376 \expandafter\global\expandafter\XeTeXmathchardef
377 \csname\string#2@sym\endcsname
378 ="\mathchar@type#3 #1 #4\relax
```

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

```
379 \cs_gset:cpn { \cs_to_str:N #2 op } {
380 \csname\string#2@sym\endcsname
381 \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
382 \ifin@
383 \expandafter\nolimits
384 \fi
385 }
```

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

```
386 \else
```

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

```
387 \expandafter\in@\expandafter#2\expandafter{\um@radicals,}
388 \ifin@
389 \cs_gset:cpn { \cs_to_str:N #2 sign } { \XeTeXradical #1 #4 \relax }
390 \else
```

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

```
391 \ifx\mathopen#3\relax
392 \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
393 \global\XeTeXdelcode#4=#1 #4\relax
```

```

394     \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
395 \else
396     \ifx\mathclose#3\relax
397         \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
398         \global\XeTeXdelcode#4=#1 #4\relax
399         \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
400     \else

```

Accents

```

401     \ifx\mathaccent#3\relax
402     \cs_gset:Npx #2 {\XeTeXmathaccent "\mathchar@type#3 #1 #4\relax}
403     \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.

```

404         \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
405     \fi
406 \fi
407 \fi
408 \fi
409 \fi
410 }

```

`\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.

```

411 \cs_set:Nn \um_set_mathcode:nnnn {
412     \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
413 }

```

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

```

414 \DeclareDocumentCommand \setmathfont { O{ } m } {

```

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

```

415     \let\glb@currsizel\relax

```


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

```
416 \let\um@char@range\@empty
417 \let\um@char@num@range\@empty
```

- Tell fontspec that maths font features are actually allowed.

```
418 \@um@fontspec@featuretrue
```

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

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

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

```
420 \def\um@mversion{normal}
421 \DeclareMathVersion{\um@mversion}
```

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

```
422 \tl_set:Nn \l_um_script_features_tl {ScriptStyle}
423 \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
424 \tl_set:Nn \l_um_script_font_tl {#2}
425 \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.

```
426 \setkeys*[um]{options}{#1}
427 \edef\@tempa{\noexpand\zf@fontspec{
428   Script = Math,
429   SizeFeatures = {
430     {Size = \tf@size-} ,
431     {Size = \sf@size-\tf@size ,
432       Font = \l_um_script_font_tl ,
433         \l_um_script_features_tl
434     } ,
435     {Size = -\sf@size ,
436       Font = \l_um_sscript_font_tl ,
437         \l_um_sscript_features_tl
438     }
439   },
440   \XKV@rm
441 }{#2}
442 }
443 \@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`:

```

444 \font\um@font="#2"\relax
445 %% \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
446 %% \um@ot@math@true
447 %% \else
448 %% \PackageWarningNoLine{unicode-math}{
449 %% The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
450 %% Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
451 %% in~ a~ substandard~ manner
452 %% }
453 %% \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

```

454 \ifx\um@char@range\@empty
455 \tl_set:Nn \um_symfont_tl {um@allsym}
456 \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
457 \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
458 \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
459 \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
460 \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
461 \else
462 \stepcounter{um@fam}
463 \tl_set:Nx \um_symfont_tl {um@fam\theum@fam}
464 \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn
465 \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
466 \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
467 \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
468 \fi

```

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

```

469 \DeclareSymbolFont{\um_symfont_tl}
470 {\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`.

```

471 \@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{b} etc.)

```

472 \um_setup_shapes:
473 \um_remap_symbols:
474 \um_setup_mathactives:
475 \um_setup_delcodes:
476 \um_setup_alphanum:
477 \um_setup_alphabets:

```

End of the `\setmathfont` macro.

```

478 }

479 \cs_new:Nn \um_setup_shapes: {
480   \um_setup_nabla:
481   \um_setup_partial:
482   \um_setup_sfshapes:
483   \um_setup_bfshapes:
484 }

```

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.

```

485 \cs_set:Nn \um_process_symbol_noparse:nnnn {
486   \um@mathsymbol{#2}{#3}{\um_symfont_t1}{#1}
487 }

488 \cs_set:Nn \um_process_symbol_parse:nnnn {
489   \um@parse@term{#1}{#2}{#3}{
490     \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
491   }
492 }

```

`\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
493 \cs_new:Nn \um_remap_symbols: {
494   \um_remap_symbol:nnn{`-}{\mathbin}{"02212}% hyphen to minus
495   \um_remap_symbol:nnn{`*}{\mathbin}{"02217}% text asterisk to "cen-
      tred asterisk"
496   \bool_if:NF \g_um_literal_colon_bool {
497     \um_remap_symbol:nnn{`:}{\mathrel}{"02236}% colon to ratio (i.e., punct to rel)
498   }

```

```

499 \if@um@literal
500   \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
501   \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
502   \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
503   \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
504 \else
505   \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um_Nabla_up_or_it_usv}
506   \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um_partial_up_or_it_u}
507 \fi

```

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

```

508 \if@um@bfliteral
509   \um_remap_symbol:nnn {\um@usv@bfNabla}{\mathord}{\um@usv@bfNabla}
510   \um_remap_symbol:nnn {\um@usv@bfitNabla}{\mathord}{\um@usv@bfitNabla}
511   \um_remap_symbol:nnn {\um@usv@bfsfNabla}{\mathord}{\um@usv@bfsfNabla}
512   \um_remap_symbol:nnn {\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsfitNabla}
513   \um_remap_symbol:nnn {\um@usv@bfpartial}{\mathord}{\um@usv@bfpartial}
514   \um_remap_symbol:nnn {\um@usv@bfitpartial}{\mathord}{\um@usv@bfitpartial}
515   \um_remap_symbol:nnn {\um@usv@bfsfpartial}{\mathord}{\um@usv@bfsfpartial}
516   \um_remap_symbol:nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
517 \else
518   \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla}{\mathord}{\um_bfNabla_up_or_it_u}
519   \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla}{\mathord}{\um_bfsfNabla_up_o}
520   \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um_bfpartial_up_o}
521   \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um_bfsfpartia}
522 \fi
523 }

```

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

```

524 \cs_new:Nn \um_remap_symbol_parse:nnn {
525   \um@parse@term {#3} {\@nil} {#2} {
526     \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
527   }
528 }
529 \cs_new:Nn \um_remap_symbol_noparse:nnn {
530   \clist_map_inline:nn {#1} {
531     \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
532   }
533 }

```

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:`

```

534 \cs_new:Nn \um_setup_mathactives: {
535   \um_make_mathactive:nNN {"2032} \primesingle \mathord
536 }

```

`\um_make_mathactive:nNN` : TODO : hook into range feature 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!

```

537 \cs_new:Nn \um_make_mathactive:nNN {
538   \XeTeXmathchardef #2 = "\mathchar@type #3
539                       \csname sym\um_symfont_tl\endcsname
540                       #1 \scan_stop:
541   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:
542 }

```

5.3.3 Delimiter codes

Some symbols that aren't mathopen/mathclose still need to have delimiter codes assigned. The list of vertical arrows may be incomplete. On the other hand, many fonts won't support them all being stretchy.

`\um_setup_delcodes:`

```

543 \cs_new:Nn \um_setup_delcodes: {
544   \um_set_delcode:nn {\`\/} {\g_um_slash_delimiter_usv}
545   \um_set_delcode:nn {"2044} {\g_um_slash_delimiter_usv} % fracslash
546   \um_set_delcode:nn {"2215} {\g_um_slash_delimiter_usv} % divslash
547   \um_set_delcode:n {"005C} % backslash
548   \um_set_delcode:nn {\`<} {"27E8} % angle brackets with ascii notation
549   \um_set_delcode:nn {\`>} {"27E9} % angle brackets with ascii notation
550   \um_set_delcode:n {"2191} % up arrow
551   \um_set_delcode:n {"2193} % down arrow
552   \um_set_delcode:n {"2195} % updown arrow
553   \um_set_delcode:n {"219F} % up arrow twohead
554   \um_set_delcode:n {"21A1} % down arrow twohead
555   \um_set_delcode:n {"21A5} % up arrow from bar
556   \um_set_delcode:n {"21A7} % down arrow from bar
557   \um_set_delcode:n {"21A8} % updown arrow from bar
558   \um_set_delcode:n {"21BE} % up harpoon right
559   \um_set_delcode:n {"21BF} % up harpoon left
560   \um_set_delcode:n {"21C2} % down harpoon right
561   \um_set_delcode:n {"21C3} % down harpoon left
562   \um_set_delcode:n {"21C5} % arrows up down
563   \um_set_delcode:n {"21F5} % arrows down up
564   \um_set_delcode:n {"21C8} % arrows up up
565   \um_set_delcode:n {"21CA} % arrows down down
566   \um_set_delcode:n {"21D1} % double up arrow
567   \um_set_delcode:n {"21D3} % double down arrow

```

```

568 \um_set_delcode:n {"21D5} % double updown arrow
569 \um_set_delcode:n {"21DE} % up arrow double stroke
570 \um_set_delcode:n {"21DF} % down arrow double stroke
571 \um_set_delcode:n {"21E1} % up arrow dashed
572 \um_set_delcode:n {"21E3} % down arrow dashed
573 }

```

`\um_setup_delcodes:` : TODO : hook into range feature

```

574 \cs_new:Nn \um_set_delcode:nn {
575   \XeTeXdelcode#1 = \csname sym\um_symfont_tl\endcsname #2
576 }
577 \cs_new:Nn \um_set_delcode:n {
578   \XeTeXdelcode#1 = \csname sym\um_symfont_tl\endcsname #1
579 }

```

5.3.4 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`.

```

580 \cs_set:Nn \um_setup_alphanum: {
581   \ifx\um@char@range\@empty
582     \um_map_chars_numbers:nn {\um@usv@num}{\um@usv@num}

```

Normal weight

```

583   \if@um@literal
584     \um_setup_literals:
585   \else
586     \um_setup_Latin:
587     \um_setup_latin:
588     \um_setup_Greek:
589     \um_setup_greek:
590   \fi

```

Bold

```

591   \if@um@bfliteral
592     \um_setup_bf_literals:
593   \else
594     \if@um@bfupLatin
595     \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfupLatin}
596     \else
597     \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
598   \fi

```

```

599     \if@um@bfuplatin
600     \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfuplatin}
601     \else
602     \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
603     \fi
604     \if@um@bfupgreek
605     \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
606     \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
607     \else
608     \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
609     \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
610     \fi
611     \if@um@bfupgreek
612     \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
613     \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
614     \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
615     \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
616     \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
617     \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
618     \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
619     \else
620     \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
621     \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
622     \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
623     \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
624     \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
625     \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
626     \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
627     \fi
628     \fi
629     \else
: TODO : what is supposed to happen here?
630     \fi
631 }

```

5.3.5 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.

```

632 \cs_set:Nn \um_mathmap_noparse:Nnn {
633   \clist_map_inline:nn {#2} {

```

```

634     \exp_args:No \um@addto@mathmap \um_symfont_t1 {##1}{#1}{#3}
635   }
636 }

\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).

637 \cs_set:Nn \um_mathmap_parse:Nnn {
638   \clist_map_inline:Nn \um@char@num@range {
639     \ifnum##1=#3\relax
640       \clist_map_inline:nn {#2} {
641         \exp_args:No \um@addto@mathmap \um_symfont_t1 {####1}{#1}{#3}
642       }
643     \fi
644   }
645 }

\um@addto@mathmap #1 : Math symbol font, always/usually the expansion of \um_symfont_t1
#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_t1 can be expanded before entering the
\g@addto@macro command.

646 \newcommand\um@addto@mathmap[4]{
647   \tl_put_right:cn {um_setup_cs_to_str:N #3:} {
648     \um_set_mathcode:nnnn{#2}{\mathalpha}{#1}{#4}
649   }
650 }

```

5.4 (Big) operators







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


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

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

USV	Ex.	Macro	Description
U+02140	$\sum_{0 \atop 1}$	<code>\Bbbsum</code>	DOUBLE-STRUCK N-ARY SUMMATION
U+0220F	$\prod_{0 \atop 1}$	<code>\prod</code>	PRODUCT OPERATOR
U+02210	$\coprod_{0 \atop 1}$	<code>\coprod</code>	COPRODUCT OPERATOR
U+02211	$\sum_{0 \atop 1}$	<code>\sum</code>	SUMMATION OPERATOR
U+0222B	\int_0^1	<code>\int</code>	INTEGRAL OPERATOR
U+0222C	\iint_0^1	<code>\iint</code>	DOUBLE INTEGRAL OPERATOR
U+0222D	\iiint_0^1	<code>\iiint</code>	TRIPLE INTEGRAL OPERATOR
U+0222E	\oint_0^1	<code>\oint</code>	CONTOUR INTEGRAL OPERATOR
U+0222F	\oiint_0^1	<code>\oiint</code>	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230	\oiint_0^1	<code>\oiint</code>	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231	\int_0^1	<code>\intclockwise</code>	CLOCKWISE INTEGRAL
U+02232	\oint_0^1	<code>\varointclockwise</code>	CONTOUR INTEGRAL, CLOCKWISE
U+02233	\oint_0^1	<code>\ointctrclockwise</code>	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0	$\bigwedge_{0 \atop 1}$	<code>\bigwedge</code>	LOGICAL OR OPERATOR
U+022C1	$\bigvee_{0 \atop 1}$	<code>\bigvee</code>	LOGICAL AND OPERATOR
U+022C2	$\bigcap_{0 \atop 1}$	<code>\bigcap</code>	INTERSECTION OPERATOR
U+022C3	$\bigcup_{0 \atop 1}$	<code>\bigcup</code>	UNION OPERATOR
U+027D5	$\Join_{0 \atop 1}$	<code>\leftouterjoin</code>	LEFT OUTER JOIN
U+027D6	$\Join_{0 \atop 1}$	<code>\rightouterjoin</code>	RIGHT OUTER JOIN
U+027D7	$\Join_{0 \atop 1}$	<code>\fullouterjoin</code>	FULL OUTER JOIN
U+027D8	$\bot_{0 \atop 1}$	<code>\bigbot</code>	LARGE UP TACK
U+027D9	$\top_{0 \atop 1}$	<code>\bigtop</code>	LARGE DOWN TACK
U+029F8	$/_{0 \atop 1}$	<code>\xsol</code>	BIG SOLIDUS
U+029F9	$\backslash_{0 \atop 1}$	<code>\xbsol</code>	BIG REVERSE SOLIDUS
U+02A00	$\odot_{0 \atop 1}$	<code>\bigodot</code>	N-ARY CIRCLED DOT OPERATOR

U+02A01	\bigoplus	<code>\bigoplus</code>	N-ARY CIRCLED PLUS OPERATOR
U+02A02	\bigotimes	<code>\bigotimes</code>	N-ARY CIRCLED TIMES OPERATOR
U+02A03	$\bigcup\cdot$	<code>\bigcupdot</code>	N-ARY UNION OPERATOR WITH DOT
U+02A04	$\bigcup+$	<code>\bigcupplus</code>	N-ARY UNION OPERATOR WITH PLUS
U+02A05	\bigcap	<code>\bigsqcap</code>	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06	\bigcup	<code>\bigsqcup</code>	N-ARY SQUARE UNION OPERATOR
U+02A07	$\&$	<code>\conjquant</code>	TWO LOGICAL AND OPERATOR
U+02A08	\vee	<code>\disjquant</code>	TWO LOGICAL OR OPERATOR
U+02A09	\times	<code>\bigtimes</code>	N-ARY TIMES OPERATOR
U+02A0B	\sum	<code>\sumint</code>	SUMMATION WITH INTEGRAL
U+02A0C	\iiint	<code>\iiiint</code>	QUADRUPLE INTEGRAL OPERATOR
U+02A0D	\int	<code>\intbar</code>	FINITE PART INTEGRAL
U+02A0E	\int	<code>\intBar</code>	INTEGRAL WITH DOUBLE STROKE
U+02A0F	\int	<code>\fint</code>	INTEGRAL AVERAGE WITH SLASH
U+02A10	\oint	<code>\cirfnint</code>	CIRCULATION FUNCTION
U+02A11	\oint	<code>\awint</code>	ANTICLOCKWISE INTEGRATION LINE INTEGRATION WITH RECTANGULAR
U+02A12	\oint	<code>\rppointint</code>	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13	\oint	<code>\scpointint</code>	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE
U+02A14	\oint	<code>\npointint</code>	POLE
U+02A15	\oint	<code>\pointint</code>	INTEGRAL AROUND A POINT OPERATOR
U+02A16	\oint	<code>\sqint</code>	QUATERNION INTEGRAL OPERATOR INTEGRAL WITH LEFTWARDS ARROW WITH
U+02A17	\int	<code>\intlarhk</code>	HOOK
U+02A18	\int	<code>\intx</code>	INTEGRAL WITH TIMES SIGN
U+02A19	\int	<code>\intcap</code>	INTEGRAL WITH INTERSECTION
U+02A1A	\int	<code>\intcup</code>	INTEGRAL WITH UNION
U+02A1B	\int	<code>\upint</code>	INTEGRAL WITH OVERBAR
U+02A1C	\int	<code>\lowint</code>	INTEGRAL WITH UNDERBAR
U+02A1D	\Join	<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.

```

651 \def\um@nolimits{
652   \@elt\int\@elt\iint\@elt\iiint\@elt\iiint\@elt\oint\@elt\oint\@elt\oiint
653   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
654   \@elt\intbar\@elt\intBar\@elt\oint\@elt\cirfnint\@elt\awint\@elt\rppoint
655   \@elt\scpoint\@elt\ntpoint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
656   \@elt\intcap\@elt\intcup\@elt\upoint\@elt\lowint
657 }

```

`\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`.

```

658 \newcommand\addnolimits[1]{
659   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
660 }

```

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

```

661 \def\removenolimits#1{
662   \begingroup
663     \def\@elt##1{
664       \ifx##1#1\else
665         \noexpand\@elt\noexpand##1
666       \fi}
667     \xdef\um@nolimits{\um@nolimits}
668   \endgroup
669 }

```

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.

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






















```
671 \let\left@primitive\left
```

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





No re-definition is made for `\right` because it's not necessary.

Here are all `\mathopen` characters:

USV	Ex.	Macro	Description
U+00028	(<code>\lparen</code>	LEFT PARENTHESIS
U+0005B	[<code>\lbrack</code>	LEFT SQUARE BRACKET
U+0007B	{	<code>\lbrace</code>	LEFT CURLY BRACKET
U+0007C		<code>\lvert</code>	VERTICAL BAR
U+02016		<code>\lVert</code>	DOUBLE VERTICAL BAR
U+0221A	√	<code>\sqrt</code>	RADICAL
U+0221B	∛	<code>\cuberoot</code>	CUBE ROOT
U+0221C	∜	<code>\fourthroot</code>	FOURTH ROOT
U+02308	⌈	<code>\lceil</code>	LEFT CEILING
U+0230A	⌊	<code>\lfloor</code>	LEFT FLOOR
U+0231C	⌒	<code>\ulcorner</code>	UPPER LEFT CORNER
U+0231E	⌔	<code>\llcorner</code>	LOWER LEFT CORNER
			LIGHT LEFT TORTOISE SHELL BRACKET
U+02772	⌞	<code>\lbrbrak</code>	ORNAMENT

U+027C5		<code>\lbag</code>	LEFT S-SHAPED BAG DELIMITER
U+027CC		<code>\longdivision</code>	LONG DIVISION MATHEMATICAL LEFT WHITE SQUARE
U+027E6		<code>\lBrack</code>	BRACKET
U+027E8		<code>\langle</code>	MATHEMATICAL LEFT ANGLE BRACKET MATHEMATICAL LEFT DOUBLE ANGLE
U+027EA		<code>\lAngle</code>	BRACKET MATHEMATICAL LEFT WHITE TORTOISE
U+027EC		<code>\Lbrbrak</code>	SHELL BRACKET
U+02983		<code>\lBrace</code>	LEFT WHITE CURLY BRACKET
U+02985		<code>\lParen</code>	LEFT WHITE PARENTHESIS
U+02987		<code>\llparenthesis</code>	Z NOTATION LEFT IMAGE BRACKET
U+02989		<code>\llangle</code>	Z NOTATION LEFT BINDING BRACKET
U+0298B		<code>\lbrackubar</code>	LEFT SQUARE BRACKET WITH UNDERBAR LEFT SQUARE BRACKET WITH TICK IN TOP
U+0298D		<code>\lbrackultick</code>	CORNER LEFT SQUARE BRACKET WITH TICK IN
U+0298F		<code>\lbracklltick</code>	BOTTOM CORNER
U+02991		<code>\langedot</code>	LEFT ANGLE BRACKET WITH DOT
U+02993		<code>\lparenless</code>	LEFT ARC LESS-THAN BRACKET
U+02997		<code>\lblkbrbrak</code>	LEFT BLACK TORTOISE SHELL BRACKET
U+029D8		<code>\lvzigzag</code>	LEFT WIGGLY FENCE
U+029DA		<code>\lvzigzag</code>	LEFT DOUBLE WIGGLY FENCE
U+029FC		<code>\lcurvyangle</code>	LEFT POINTING CURVED ANGLE BRACKET
U+03014		<code>\lbrbrak</code>	LEFT BROKEN BRACKET
U+03018		<code>\Lbrbrak</code>	LEFT WHITE TORTOISE SHELL BRACKET

And `\mathclose`:

USV	Ex.	Macro	Description
U+00029)	<code>\rparen</code>	RIGHT PARENTHESIS
U+0005D]	<code>\rbrack</code>	RIGHT SQUARE BRACKET
U+0007C		<code>\rvert</code>	VERTICAL BAR
U+0007D	}	<code>\rbrace</code>	RIGHT CURLY BRACKET
U+02016		<code>\rVert</code>	DOUBLE VERTICAL BAR
U+02309]̂	<code>\rceil</code>	RIGHT CEILING
U+0230B]̇	<code>\rfloor</code>	RIGHT FLOOR
U+0231D	┐	<code>\urcorner</code>	UPPER RIGHT CORNER
U+0231F	└	<code>\lrcorner</code>	LOWER RIGHT CORNER LIGHT RIGHT TORTOISE SHELL BRACKET
U+02773		<code>\rbrbrak</code>	ORNAMENT
U+027C6		<code>\rbag</code>	RIGHT S-SHAPED BAG DELIMITER MATHEMATICAL RIGHT WHITE SQUARE
U+027E7		<code>\rBrack</code>	BRACKET
U+027E9		<code>\rangle</code>	MATHEMATICAL RIGHT ANGLE BRACKET

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

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	ˆ	\acute	ACUTE ACCENT
U+00302	ˆ	\hat	CIRCUMFLEX ACCENT
U+00303	˜	\tilde	TILDE
U+00304	ˉ	\bar	MACRON
U+00305	ˆ	\overbar	OVERBAR EMBELLISHMENT
U+00306	˘	\breve	BREVE
U+00307	˙	\dot	DOT ABOVE
U+00308	¨	\ddot	DIERESIS
U+00309	ˆ	\ovhook	COMBINING HOOK ABOVE
U+0030A	ˆ	\ocirc	RING
U+0030C	ˇ	\check	CARON
U+00310	ˆ	\candra	CANDRABINDU (NON-SPACING)
U+00312	ˆ	\turnedcomma	COMBINING TURNED COMMA ABOVE GREEK PSILI (SMOOTH BREATHING)
U+00313	ˆ	\smooth	(NON-SPACING)

U+00314	\dot{x}	<code>\orough</code>	GREEK DASIA (ROUGH BREATHING) (NON-SPACING)
U+00315	\dot{x}	<code>\ocommatopright</code>	COMBINING COMMA ABOVE RIGHT
U+0031A	\dot{x}	<code>\droang</code>	LEFT ANGLE ABOVE (NON-SPACING) COMBINING LONG SOLIDUS OVERLAY
U+00338	$\not x$	<code>\not</code>	
U+020D0	\overleftarrow{x}	<code>\leftharpoonaccent</code>	COMBINING LEFT HARPOON ABOVE
U+020D1	\overrightarrow{x}	<code>\rightharpoonaccent</code>	COMBINING RIGHT HARPOON ABOVE
U+020D2	\overline{x}	<code>\vertoverlay</code>	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6	\overleftarrow{x}	<code>\overleftarrow</code>	COMBINING LEFT ARROW ABOVE
U+020D7	\overrightarrow{x}	<code>\overrightarrow</code>	COMBINING RIGHT ARROW ABOVE
U+020DB	\ddot{x}	<code>\dddot</code>	COMBINING THREE DOTS ABOVE
U+020DC	\ddot{x}	<code>\ddddot</code>	COMBINING FOUR DOTS ABOVE
U+020E1	\overleftrightarrow{x}	<code>\overleftrightharpoon</code>	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7	$\text{\textcircled{x}}$	<code>\annuity</code>	COMBINING ANNUITY SYMBOL
U+020E8	$\text{\textcircled{x}}$	<code>\threeunderdot</code>	COMBINING TRIPLE UNDERDOT
U+020E9	$\text{\textcircled{x}}$	<code>\widebridgeabove</code>	COMBINING WIDE BRIDGE ABOVE COMBINING RIGHTWARDS HARPOON WITH
U+020EC	$\text{\textcircled{x}}$	<code>\underrightharpoonowdown</code>	BARB DOWNWARDS COMBINING LEFTWARDS HARPOON WITH
U+020ED	$\text{\textcircled{x}}$	<code>\underleftharpoonowdown</code>	BARB DOWNWARDS
U+020EE	$\text{\textcircled{x}}$	<code>\underleftarrow</code>	COMBINING LEFT ARROW BELOW
U+020EF	$\text{\textcircled{x}}$	<code>\underrightarrow</code>	COMBINING RIGHT ARROW BELOW
U+020F0	$\text{\textcircled{x}}$	<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 commands.

```

673 \newcommand\um@zf@feature[2]{
674   \define@key[zf]{options}{#1}[]{}
675   \if@um@fontspec@feature
676     #2
677   \else
678     \PackageError{fontspec/unicode-math}
679     {The '#1' font feature can only be used for maths fonts}
680     {The feature you tried to use can only be in commands
681      like \protect\setmathfont}
682   \fi
683 }
684 }
```

6.1 OpenType maths font features

```

685 \um@zf@feature{ScriptStyle}{
686   \zf@update@ff{+ssty=0}
687 }
688 \um@zf@feature{ScriptScriptStyle}{
689   \zf@update@ff{+ssty=1}
690 }

```

6.2 Script and scriptscript font options

```

691 \define@cmdkey[um]{options}[um@]{ScriptFeatures}{}
692 \define@cmdkey[um]{options}[um@]{ScriptScriptFeatures}{}
693 \define@cmdkey[um]{options}[um@]{ScriptFont}{}
694 \define@cmdkey[um]{options}[um@]{ScriptScriptFont}{}

```

6.3 Range processing

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

```

695 \define@choicekey+[um]{options}{Range}[\@tempa\@tempb]{ALL}{
696   \ifcase\@tempb\relax
697   \global\let\um@char@range\@empty
698   \fi
699 }{
700   \xdef\um@char@range{#1}
701 }

```

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.

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

```

702 \newcommand\um@parse@term[4]{
703   \clist_map_variable:NNn \um@char@range \@ii {
704     \unless\ifx\@ii\@empty
705       \@tempswafalse

```


Table 11: Ranges accepted by `\um@parse@range`.

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

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

```

706     \expandafter\um@firstchar\expandafter{\@ii}
707     \ifx\@tempa\um@backslash
708         \expandafter\ifx\@ii#2\relax
709             \@tempswatrue
710         \else
711             \expandafter\ifx\@ii#3\relax
712                 \@tempswatrue
713             \fi
714         \fi

```

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

```

715     \else
716         \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
717     \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.

```

718     \if@tempswa
719         \ifx\um@char@num@range\@empty
720             \g@addto@macro\um@char@num@range{#1}
721         \else
722             \g@addto@macro\um@char@num@range{, #1}
723         \fi
724         #4%
725     \fi
726 \fi
727 }
728 }
729 \def\um@firstof#1#2\@nil{#1}
730 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
731 \def\um@firstchar#1{\edef\@tempa{\expandafter\um@firstof\string#1\@nil}}

```

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

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

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

732	<code>\def\um@parse@range#1-#2-#3\@nil#4\@nil{</code>
733	<code>\def\@tempa{#1}</code>
734	<code>\def\@tempb{#2}</code>
	Range $r = x$
	C-list input <code>\@ii=X</code>
	Macro input <code>\um@parse@range X-\@marker-\@nil#1\@nil</code>
	Arguments $\#1-\#2-\#3 = X-\@marker-\{ \}$
735	<code>\expandafter\ifx\expandafter\@marker\@tempb\relax</code>
736	<code>\ifnum#4=#1\relax</code>
737	<code>\@tempswatrue</code>
738	<code>\fi</code>
739	<code>\else</code>
	Range $r \geq x$
	C-list input <code>\@ii=X-</code>
	Macro input <code>\um@parse@range X--\@marker-\@nil#1\@nil</code>
	Arguments $\#1-\#2-\#3 = X-\{ \}-\@marker-$
740	<code>\ifx\@empty\@tempb</code>
741	<code>\ifnum#4>\numexpr#1-1\relax</code>
742	<code>\@tempswatrue</code>
743	<code>\fi</code>
744	<code>\else</code>
	Range $r \leq y$
	C-list input <code>\@ii=-Y</code>
	Macro input <code>\um@parse@range -Y-\@marker-\@nil#1\@nil</code>
	Arguments $\#1-\#2-\#3 = \{ \}-Y-\@marker-$
745	<code>\ifx\@empty\@tempa</code>
746	<code>\ifnum#4<\numexpr#2+1\relax</code>
747	<code>\@tempswatrue</code>
748	<code>\fi</code>
	Range $x \leq r \leq y$
	C-list input <code>\@ii=X-Y</code>
	Macro input <code>\um@parse@range X-Y-\@marker-\@nil#1\@nil</code>
	Arguments $\#1-\#2-\#3 = X-Y-\@marker-$

```

749     \else
750     \ifnum#4>\numexpr#1-1\relax
751     \ifnum#4<\numexpr#2+1\relax
752     \@tempwattrue
753     \fi
754     \fi
755     \fi
756     \fi
757     \fi
758 }

\um_map_char:nn #1 : Number of iterations
                #2 : Starting input char(s)
                #3 : Starting output char
                Loops through character ranges setting \mathcode.
759 \cs_set:Nn \um_map_chars_range:nnn {
760   \clist_map_variable:nNn {#2} \l_um_input_num {
761     \prg_stepwise_variable:nnnNn{0}{1}{#1} \l_um_incr_num {
762       \um_set_mathcode:nnnn
763       {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
764       {\mathalpha}{\um_symfont_t1}
765       {\numexpr \l_um_incr_num + #3 \relax}
766     }
767   }
768 }
769 \cs_set:Nn \um_map_chars_latin:nn {
770   \um_map_chars_range:nnn {25}{#1}{#2}
771 }
772 \cs_set:Nn \um_map_chars_greek:nn {
773   \um_map_chars_range:nnn {24}{#1}{#2}
774 }
775 \cs_set:Nn \um_map_chars_numbers:nn {
776   \um_map_chars_range:nnn {9}{#1}{#2}
777 }
778 \cs_set:Nn \um_map_char:nn {
779   \um_map_chars_range:nnn {0}{#1}{#2}
780 }

```

```

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

```

```

785     {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
786   }
787 }

\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.

788 \cs_new:Nn \um_set_mathalph_range:nNnn {
789   \clist_map_variable:nNn {#3} \l_um_input_num {
790     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
791       \exp_args:Nnff \um_mathmap:Nnn {#2}
792       {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
793       {\number\numexpr \l_um_inc_num + #4 \relax}
794     }
795   }
796 }
797 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
798   \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
799 }
800 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
801   \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
802 }
803 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
804   \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
805 }



---


BCDBCDEABCDEFG \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.

```

806 \AtBeginDocument{\um@resolve@greek}
807 \newcommand\um@resolve@greek{
808   \def\Alpha{\mitAlpha}
809   \def\Beta{\mitBeta}
810   \def\Gamma{\mitGamma}
811   \def\Delta{\mitDelta}
812   \def\Epsilon{\mitEpsilon}

```

```

813 \def\Zeta{\mitZeta}
814 \def\Eta{\mitEta}
815 \def\Theta{\mitTheta}
816 \def\Iota{\mitIota}
817 \def\Kappa{\mitKappa}
818 \def\Lambda{\mitLambda}
819 \def\Mu{\mitMu}
820 \def\Nu{\mitNu}
821 \def\Xi{\mitXi}
822 \def\Omicron{\mitOmicron}
823 \def\Pi{\mitPi}
824 \def\Rho{\mitRho}
825 \def\varTheta{\mitvarTheta}
826 \def\Sigma{\mitSigma}
827 \def\Tau{\mitTau}
828 \def\Upsilon{\mitUpsilon}
829 \def\Phi{\mitPhi}
830 \def\Chi{\mitChi}
831 \def\Psi{\mitPsi}
832 \def\Omega{\mitOmega}

```

Lowercase:

```

833 \def\alpha{\mitalpha}
834 \def\beta{\mitbeta}
835 \def\gamma{\mitgamma}
836 \def\delta{\mitdelta}
837 \def\epsilon{
838   \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
839 }
840 \def\zeta{\mitzeta}
841 \def\eta{\miteta}
842 \def\theta{\mittheta}
843 \def\iota{\mitiota}
844 \def\kappa{\mitkappa}
845 \def\lambda{\mitlambda}
846 \def\mu{\mitmu}
847 \def\nu{\mitnu}
848 \def\xi{\mitxi}
849 \def\omicron{\mitomicron}
850 \def\pi{\mitpi}
851 \def\rho{\mitrho}
852 \def\varsigma{\mitvarsigma}
853 \def\sigma{\mitsigma}
854 \def\tau{\mittau}
855 \def\upsilon{\mitupsilon}
856 \def\phi{
857   \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}

```

```

858 }
859 \def\chi{\mitchi}
860 \def\psi{\mitpsi}
861 \def\omega{\mitomega}
862 \def\varepsilon{
863   \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
864 }
865 \def\vartheta{\mitvartheta}
866 \def\varkappa{\mitvarkappa}
867 \def\varphi{
868   \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
869 }
870 \def\varrho{\mitvarrho}
871 \def\varpi{\mitvarpi}
872 }

```

6.5 Setting up the mappings

\um_setup_literals: : TODO : other literal symbols

```

873 \cs_set:Nn \um_setup_literals: {
874   \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}
875   \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}
876   \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}
877   \um_map_char:nn {\um@usv@ith}{\um@usv@ith}
878   \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}
879   \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}
880   \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}
881   \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}
882   \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}
883 }

```

\um_setup_bf_literals: TODO: other literal symbols

```

884 \cs_set:Nn \um_setup_bf_literals: {
885   \um_map_chars_latin:nn {\um@usv@bfupLatin}{\um@usv@bfupLatin}
886   \um_map_chars_latin:nn {\um@usv@bfuplatin}{\um@usv@bfuplatin}
887   \um_map_chars_latin:nn {\um@usv@bfitLatin}{\um@usv@bfitLatin}
888   \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}
889   \um_map_chars_greek:nn {\um@usv@bfupGreek}{\um@usv@bfupGreek}
890   \um_map_chars_greek:nn {\um@usv@bfupgreek}{\um@usv@bfupgreek}
891   \um_map_chars_greek:nn {\um@usv@bfitGreek}{\um@usv@bfitGreek}
892   \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
893 }

```

\um_setup_Latin:

```

894 \cs_set:Nn \um_setup_Latin: {
895   \if@um@upLatin

```

```

896 \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
897 \else
898 \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
899 \fi
900 }

```

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

```

901 \cs_set:Nn \um_setup_latin: {
902   \if@um@uplatin
903     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
904     \um_map_char:nn {\um@usv@ith}{`\h}
905   \else
906     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
907     \um_map_char:nn {\`\h,\um@usv@ith}{\um@usv@ith}
908   \fi
909 }

```

\um_setup_greek:

```

910 \cs_set:Nn \um_setup_greek: {
911   \if@um@upgreek
912     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
913     \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
914   \else
915     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
916     \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}
917   \fi
918 }

```

\um_setup_greek:

```

919 \cs_set:Nn \um_setup_greek: {
920   \if@um@upgreek
921     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
922     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
923     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
924     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
925     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
926     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
927     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
928   \else
929     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
930     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
931     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
932     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
933     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
934     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}

```

```

935 \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
936 \fi
937 }

```

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.

```

938 \cs_new:Nn \um_setup_alphabets: {
939   \um_setup_math_alphabet:nn {up}   {\latin,Latin,greek,Greek}
940   \um_setup_math_alphabet:nn {it}   {\latin,Latin,greek,Greek}
941   \um_setup_math_alphabet:nn {bb}   {\latin,Latin,num}
942   \um_setup_math_alphabet:nn {scr}  {\latin,Latin}
943   \um_setup_math_alphabet:nn {frak} {\latin,Latin}
944   \um_setup_math_alphabet:nn {sf}   {\latin,Latin,num}
945   \um_setup_math_alphabet:nn {sfup} {\latin,Latin,num}
946   \um_setup_math_alphabet:nn {ffit} {\latin,Latin,num}
947   \um_setup_math_alphabet:nn {tt}   {\latin,Latin,num}
948   \um_setup_math_alphabet:nn {bf}   {\latin,Latin,greek,Greek,num}
949   \um_setup_math_alphabet:nn {bfup} {\latin,Latin,greek,Greek,num}
950   \um_setup_math_alphabet:nn {bfit} {\latin,Latin,greek,Greek,num}
951   \um_setup_math_alphabet:nn {bfscr} {\latin,Latin}
952   \um_setup_math_alphabet:nn {bffrak}{latin,Latin}

```



```

953 \um_setup_math_alphabet:nn {bfsf }{latin, Latin, greek, Greek, num}
954 \um_setup_math_alphabet:nn {bfsfup}{latin, Latin, greek, Greek, num}
955 \um_setup_math_alphabet:nn {bfsfit}{latin, Latin, greek, Greek, num}
956 }

```

`\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.

```

957 \cs_new:Nn \um_setup_math_alphabet:nn {
958   \clist_map_inline:nn {#2} {
959     \um_glyph_if_exist:nT {\csname um@usv@#1##1 \endcsname}{
960       \um_maybe_init_alphabet:n {#1}
961       \um_prepare_alph:n {#1}
962       \clist_map_break:
963     }
964   }
965   \clist_map_inline:nn {#2} {
966     \um_glyph_if_exist:nTF {\csname um@usv@#1##1 \endcsname}{
967       \use:c {um_config_math#1_##1:}
968     }{
969       \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
970         Math~ alphabet~
971         \@backslashchar math#1~
972         (\tl_use:c{g_um_math_alphabet_name_#1_tl})~
973         not~ found~ in~ font~
974         \fontname\um@font}
975     }
976   }
977 }
978 \tl_set:Nn \g_um_math_alphabet_name_latin_tl {Latin, lowercase}
979 \tl_set:Nn \g_um_math_alphabet_name_Latin_tl {Latin, uppercase}
980 \tl_set:Nn \g_um_math_alphabet_name_greek_tl {Greek, lowercase}
981 \tl_set:Nn \g_um_math_alphabet_name_Greek_tl {Greek, uppercase}
982 \tl_set:Nn \g_um_math_alphabet_name_num_tl {Numerals}

983 \cs_set:Nn \um_init_alphabet:n {
984   \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
985 }

```

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

```

986 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
987   \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
988 }

```

`\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.

```

989 \cs_new:Nn \um_prepare_alph:n {
990   \cs_if_exist:cF {um_math#1:n} {
991     \cs_set:cpn {um_math#1:n} ##1 {
992       \use:c {um_setup_math#1:} ##1 \egroup
993     }
994     \cs_set_protected:cpn {math#1} {
995       \bgroup
996       \mode_if_math:F {
997         \egroup\expandafter
998         \non@alpherr\expandafter{\csname math#1\endcsname\space}
999       }
1000       \use:c {um_math#1:n}
1001     }
1002   }
1003 }

```

: 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.

```

1004 \cs_new:Npn \um_config_mathup_Latin: {
1005   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
1006 }
1007 \cs_new:Npn \um_config_mathup_latin: {
1008   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
1009   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@ith} {\` \h}
1010 }
1011 \cs_new:Npn \um_config_mathup_Greek: {
1012   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
1013   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}
1014   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varThe}
1015 }
1016 \cs_new:Npn \um_config_mathup_greek: {
1017   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
1018   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}
1019   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@va}
1020   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@varthe}
1021   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkap}
1022   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
1023   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
1024   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}

```

1025 }

7.1.2 Italic: `\mathit`

```
1026 \cs_new:Npn \um_config_mathit_Latin: {
1027   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
1028 }
1029 \cs_new:Npn \um_config_mathit_latin: {
1030   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
1031   \um_set_mathalphabet_char:Nnn{\mathit}{`\h,\um@usv@ith}{\um@usv@ith}
1032 }
1033 \cs_new:Npn \um_config_mathit_Greek: {
1034   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
1035   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarTheta}
1036   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
1037 }
1038 \cs_new:Npn \um_config_mathit_greek: {
1039   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
1040   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartial}
1041   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
1042   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
1043   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
1044   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
1045   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
1046   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
1047 }
```

7.1.3 Blackboard or double-struck: `\mathbb`

```
1048 \cs_new:Npn \um_config_mathbb_latin: {
1049   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}
1050   \um_set_mathalphabet_char:Nnn{\mathbb}{\um@usv@ith} {"1D559}
1051 }
1052 \cs_new:Npn \um_config_mathbb_Latin: {
1053   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
1054   \um_set_mathalphabet_char:Nnn{\mathbb}{`\C,"1D436}{ "2102}
1055   \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D43B}{ "210D}
1056   \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D441}{ "2115}
1057   \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D443}{ "2119}
1058   \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D444}{ "211A}
1059   \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D445}{ "211D}
1060   \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D44D} {"2124}
1061 }
1062 \cs_new:Npn \um_config_mathbb_num: {
1063   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@num}{\um@usv@bbnum}
1064 }
```

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

```
1065 \cs_new:Npn \um_config_mathscr_Latin: {  
1066   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}  
1067   \um_set_mathalphabet_char:Nnn \mathscr {\` \B,"1D435}{ "212C}  
1068   \um_set_mathalphabet_char:Nnn \mathscr {\` \E,"1D438}{ "2130}  
1069   \um_set_mathalphabet_char:Nnn \mathscr {\` \F,"1D439}{ "2131}  
1070   \um_set_mathalphabet_char:Nnn \mathscr {\` \H,"1D43B}{ "210B}  
1071   \um_set_mathalphabet_char:Nnn \mathscr {\` \I,"1D43C}{ "2110}  
1072   \um_set_mathalphabet_char:Nnn \mathscr {\` \L,"1D43F}{ "2112}  
1073   \um_set_mathalphabet_char:Nnn \mathscr {\` \M,"1D440}{ "2133}  
1074   \um_set_mathalphabet_char:Nnn \mathscr {\` \R,"1D445}{ "211B}  
1075 }  
1076 \cs_new:Npn \um_config_mathscr_latin: {  
1077   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrlatin}  
1078   \um_set_mathalphabet_char:Nnn \mathscr {\` \e,"1D452}{ "212F}  
1079   \um_set_mathalphabet_char:Nnn \mathscr {\` \g,"1D454}{ "210A}  
1080   \um_set_mathalphabet_char:Nnn \mathscr {\` \o,"1D45C}{ "2134}  
1081   \um_set_mathalphabet_char:Nnn \mathscr {\um@usv@ith} { "1D4BD}  
1082 }
```

7.1.5 Fraktur or fraktur or blackletter: `\mathfrak`

```
1083 \cs_new:Npn \um_config_mathfrak_Latin: {  
1084   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLat  
1085   \um_set_mathalphabet_char:Nnn \mathfrak {\` \C,"1D436}{ "212D}  
1086   \um_set_mathalphabet_char:Nnn \mathfrak {\` \H,"1D43B}{ "210C}  
1087   \um_set_mathalphabet_char:Nnn \mathfrak {\` \I,"1D43C}{ "2111}  
1088   \um_set_mathalphabet_char:Nnn \mathfrak {\` \R,"1D445}{ "211C}  
1089   \um_set_mathalphabet_char:Nnn \mathfrak {\` \Z,"1D44D}{ "2128}  
1090 }  
1091 \cs_new:Npn \um_config_mathfrak_latin: {  
1092   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklati  
1093   \um_set_mathalphabet_char:Nnn \mathfrak {\um@usv@ith} { "1D525}  
1094 }
```

7.1.6 Sans serif: `\mathsf`

```
1095 \cs_new:Npn \um_config_mathsf_Latin: {  
1096   \bool_if:NTF \g_um_sf literal_bool {  
1097     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin}{\um@usv@sfulatin}  
1098     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itLatin}{\um@usv@sfitLatin}  
1099   }{  
1100     \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@upLatin,\um@usv@itLatin}{ \um_sf_Latin_up  
1101   }  
1102 }  
1103 \cs_new:Npn \um_config_mathsf_latin: {  
1104   \bool_if:NTF \g_um_sf literal_bool {  
1105     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin}{\um@usv@sfulatin}
```

```

1106 \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itlatin}{\um@usv@sfitlatin}
1107 \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}
1108 }{
1109 \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@uplatin,\um@usv@itlatin}{ \um_sf_latin_up}
1110 \bool_if:NTF \g_um_upsans_bool {
1111 \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D5C1}
1112 }{
1113 \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}
1114 }
1115 }
1116 }
1117 \cs_new:Npn \um_config_mathsf_num: {
1118 \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnun}
1119 }

```

7.1.7 Sans serif upright: `\mathsfup`

```

1120 \cs_new:Npn \um_config_mathsfup_num: {
1121 \um_set_mathalphabet_numbers:Nnn{\mathsfup}{\um@usv@num}{\um@usv@sfnun}
1122 }
1123 \cs_new:Npn \um_config_mathsfup_latin: {
1124 \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfulat}
1125 \um_set_mathalphabet_char:Nnn \mathsfup {\um@usv@ith} {"1D5C1}
1126 }
1127 \cs_new:Npn \um_config_mathsfup_Latin: {
1128 \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfulat}
1129 }

```

7.1.8 Sans serif italic: `\mathsfit`

Map the numbers like that because it seems sensible.

```

1130 \cs_new:Npn \um_config_mathsfitt_num: {
1131 \um_set_mathalphabet_numbers:Nnn{\mathsfit}{\um@usv@num}{\um@usv@sfnun}
1132 }
1133 \cs_new:Npn \um_config_mathsfitt_Latin: {
1134 \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sffitLat}
1135 }
1136 \cs_new:Npn \um_config_mathsfitt_latin: {
1137 \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sffitlat}
1138 \um_set_mathalphabet_char:Nnn \mathsfit {\um@usv@ith} {"1D629}
1139 }

```

7.1.9 Typewriter or monospaced: `\mathtt`

```

1140 \cs_new:Npn \um_config_mathtt_num: {
1141 \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
1142 }
1143 \cs_new:Npn \um_config_mathtt_Latin: {

```

```

1144 \um_set_mathalphabet_latin:Nnn{\mathhtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
1145 }
1146 \cs_new:Npn \um_config_mathhtt_latin: {
1147 \um_set_mathalphabet_latin:Nnn{\mathhtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
1148 \um_set_mathalphabet_char:Nnn \mathhtt {\um@usv@ith} {"1D691}
1149 }

```

7.2 Bold math alphabets

7.2.1 Bold: $\mathbf{}$

```

1150 \cs_new:Npn \um_config_mathbf_num: {
1151 \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
1152 }
1153 \cs_new:Npn \um_config_mathbf_Latin: {
1154 \if@um@bfliteral
1155 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfupLatin}
1156 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
1157 \else
1158 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bf_Latin_up_}
1159 \fi
1160 }
1161 \cs_new:Npn \um_config_mathbf_latin: {
1162 \if@um@bfliteral
1163 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfupLatin}
1164 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
1165 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1166 \else
1167 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bf_Latin_up_}
1168 \if@um@bfupLatin
1169 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
1170 \else
1171 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1172 \fi
1173 \fi
1174 }
1175 \cs_new:Npn \um_config_mathbf_Greek: {
1176 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
1177 \if@um@bfliteral
1178 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfupGreek}
1179 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
1180 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
1181 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
1182 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
1183 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
1184 \else
1185 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bf_Greek_up_}
1186 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla_up_or_i}

```

```

1187     \if@um@bfupgreek
1188     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarTheta}
1189     \else
1190     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarTheta}
1191     \fi
1192 \fi
1193 }
1194 \cs_new:Npn \um_config_mathbf_greek: {
1195   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
1196   \if@um@bfliteral
1197     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfupgreek}
1198     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitgreek}
1199     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
1200     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bfvarepsilon}
1201     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
1202     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
1203     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
1204     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
1205     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}
1206     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
1207     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
1208     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
1209     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1210     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
1211     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
1212     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
1213   \else
1214     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bf_greek_up}
1215     \if@um@bfupgreek
1216       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
1217       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1218       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1219       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1220       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1221       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1222     \else
1223       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
1224       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1225       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1226       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1227       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1228       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1229     \fi
1230     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartial}
1231   \fi
1232 }

```

7.2.2 Bold Italic: `\mathbfit`

```
1233 \cs_new:Npn \um_config_mathbfit_num: {  
1234   \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnum}  
1235 }  
1236 \cs_new:Npn \um_config_mathbfit_Latin: {  
1237   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLat  
1238 }  
1239 \cs_new:Npn \um_config_mathbfit_latin: {  
1240   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlat  
1241   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@ith} {"1D489}  
1242 }  
1243 \cs_new:Npn \um_config_mathbfit_Greek: {  
1244   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGre  
1245   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfit  
1246   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}  
1247 }  
1248 \cs_new:Npn \um_config_mathbfit_greek: {  
1249   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLat  
1250   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgre  
1251   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpa  
1252   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@  
1253   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfit  
1254   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfit  
1255   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarp  
1256   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarr  
1257   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}  
1258 }
```

7.2.3 Bold Italic: `\mathbfup`

```
1259 \cs_new:Npn \um_config_mathbfup_num: {  
1260   \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnum}  
1261 }  
1262 \cs_new:Npn \um_config_mathbfup_Latin: {  
1263   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLat  
1264 }  
1265 \cs_new:Npn \um_config_mathbfup_latin: {  
1266   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplat  
1267   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@ith} {"1D421}  
1268 }  
1269 \cs_new:Npn \um_config_mathbfup_Greek: {  
1270   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGre  
1271   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfva  
1272   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}  
1273   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Digamma}{\um@usv@bfDigamma}  
1274 }  
1275 \cs_new:Npn \um_config_mathbfup_greek: {
```



```

1276 \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgre
1277 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpart
1278 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
1279 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfva
1280 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfva
1281 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi
1282 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho
1283 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1284 \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@digamma}{\um@usv@bfdigamma}
1285 }

```

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

```

1286 \cs_new:Npn \um_config_mathbffrak_Latin: {
1287   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bffra
1288 }
1289 \cs_new:Npn \um_config_mathbffrak_latin: {
1290   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bffra
1291   \um_set_mathalphabet_char:Nnn{\mathbffrak}{\um@usv@ith} {"1D58D}
1292 }

```

7.2.5 Bold script or calligraphic: `\mathbfscr`

```

1293 \cs_new:Npn \um_config_mathbfscr_Latin: {
1294   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrL
1295 }
1296 \cs_new:Npn \um_config_mathbfscr_latin: {
1297   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrL
1298   \um_set_mathalphabet_char:Nnn{\mathbfscr}{\um@usv@ith} {"1D4F1}
1299 }

```

7.2.6 Bold sans serif: `\mathbfsf`

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

```

1300 \cs_new:Npn \um_config_mathbfsf_num: {
1301   \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1302 }
1303 \cs_new:Npn \um_config_mathbfsf_Latin: {
1304   \bool_if:NTF \g_um_sfliteral_bool {
1305     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin}{\um@usv@bfsfupLatin}
1306     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itLatin}{\um@usv@bfsfitLatin}
1307   }{
1308     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsf_Lati
1309   }
1310 }
1311 \cs_new:Npn \um_config_mathbfsf_latin: {
1312   \bool_if:NTF \g_um_sfliteral_bool {
1313     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@uplatin}{\um@usv@bfsfuplatin}

```

```

1314 \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itlatin}{\um@usv@bfsfitlatin}
1315 \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
1316 }{
1317 \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um_bfsf_lati
1318 \bool_if:NTF \g_um_upsans_bool {
1319 \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D5F5}
1320 }{
1321 \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
1322 }
1323 }
1324 }
1325 \cs_new:Npn \um_config_mathbfsf_Greek: {
1326 \bool_if:NTF \g_um_sfliteral_bool {
1327 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek}{\um@usv@bfsfupGreek}
1328 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itGreek}{\um@usv@bfsfitGreek}
1329 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{ "1D767}
1330 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{ "1D7A1}
1331 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla}{\um@usv@bfsfNabla}
1332 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itNabla}{\um@usv@bfsfitNabla}
1333 }{
1334 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um_bfsf_Gree
1335 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{\um_bfsfNabla_up
1336 \bool_if:NTF \g_um_upsans_bool {
1337 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{ "1D767}
1338 }{
1339 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{ "1D7A1}
1340 }
1341 }
1342 }
1343 \cs_new:Npn \um_config_mathbfsf_greek: {
1344 \bool_if:NTF \g_um_sfliteral_bool {
1345 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek}{\um@usv@bfsfupgreek}
1346 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itgreek}{\um@usv@bfsfitgreek}
1347 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial}{\um@usv@bfsfpartial}
1348 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon}{ "1D78A}
1349 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta}{ "1D78B}
1350 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa}{ "1D78C}
1351 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi}{ "1D78D}
1352 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho}{ "1D78E}
1353 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi}{ "1D78F}
1354 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itpartial}{\um@usv@bfsfitpartial}
1355 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarepsilon}{ "1D7C4}
1356 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvartheta}{ "1D7C5}
1357 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarkappa}{ "1D7C6}
1358 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarphi}{ "1D7C7}
1359 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarrho}{ "1D7C8}

```

```

1360 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarpi}{1D7C9}
1361 }{
1362 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um_bfsf_gree
1363 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial,\um@usv@itpartial}{\um_bfsfpart
1364 \bool_if:NTF \g_um_upsans_bool {
1365 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7
1366 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1367 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1368 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1369 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{1D78E}
1370 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1371 }{
1372 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7
1373 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{1D7C5}
1374 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{1D7C6}
1375 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{1D7C7}
1376 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{1D7C8}
1377 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{1D7C9}
1378 }
1379 }
1380 }

```

7.2.7 Bold upright sans serif: \mathbfsfup

```

1381 \cs_new:Npn \um_config_mathbfsfup_num: {
1382 \um_set_mathalphabet_numbers:Nnn{\mathbfsfup}{\um@usv@num}{\um@usv@bfsfnum}
1383 }
1384 \cs_new:Npn \um_config_mathbfsfup_Latin: {
1385 \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfu
1386 }
1387 \cs_new:Npn \um_config_mathbfsfup_latin: {
1388 \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfu
1389 \um_set_mathalphabet_char:Nnn \mathbfsfup {\um@usv@ith} {1D5F5}
1390 }
1391 \cs_new:Npn \um_config_mathbfsfup_Greek: {
1392 \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfu
1393 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varTheta,\um@usv@itvarTheta}{1D767}
1394 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@Nabla,\um@usv@itNabla}{1D76F}
1395 }
1396 \cs_new:Npn \um_config_mathbfsfup_greek: {
1397 \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfu
1398 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@partial,\um@usv@itpartial}{1D789}
1399 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D78A}
1400 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1401 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1402 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1403 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varrho,\um@usv@itvarrho}{1D78E}

```

```

1404 \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1405 }

```

7.2.8 Bold italic sans serif: `\mathbfsfit`

```

1406 \cs_new:Npn \um_config_mathbfsfit_num: {
1407   \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@num}{\um@usv@bfsfnum}
1408 }
1409 \cs_new:Npn \um_config_mathbfsfit_Latin: {
1410   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfi
1411 }
1412 \cs_new:Npn \um_config_mathbfsfit_latin: {
1413   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfi
1414   \um_set_mathalphabet_char:Nnn \mathbfsfit {\um@usv@ith} {1D65D}
1415 }
1416 \cs_new:Npn \um_config_mathbfsfit_Greek: {
1417   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfi
1418   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{1D7A1}
1419   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNa
1420 }
1421 \cs_new:Npn \um_config_mathbfsfit_greek: {
1422   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfi
1423   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsf
1424   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7C4}
1425   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{1D7C5}
1426   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{1D7C6}
1427   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{1D7C7}
1428   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{1D7C8}
1429   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{1D7C9}
1430 }

```

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.

```

1431 \begingroup
1432   \char_make_other:N \^
1433   \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1434     \lowercase{

```

```

1435     \scantokens{\global\let#1=^^^^^#2}
1436   }
1437 }

```

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

```

1438   \gdef\um@scanactivedef"#1\@nil#2{
1439     \lowercase{
1440       \tl_rescan:nn{
1441         \ExplSyntaxOn
1442         \char_make_math_superscript:N\^
1443       }{
1444         \global\def^^^^^#1{#2}
1445       }
1446     }
1447   }
1448 \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.

```

1449 \begingroup
1450 \def\UnicodeMathSymbol#1#2#3#4{
1451   \um@scancharlet#2=#1\@nil
1452 }
1453 \char_make_other:N \#
1454 \@input{unicode-math-table.tex}
1455 \endgroup

```

Fix \backslash:

```

1456 \group_begin:
1457   \lccode`\*=`\
1458   \char_make_escape:N \
1459   \char_make_other:N \
1460   |lowercase{
1461   |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 (`\primetriple`): x'''
 U+2057: QUADRUPLE PRIME (`\primequadruple`): x''''

As you can see, they're all drawn at the correct height without being superscripted. However, in a correctly behaviour OpenType font with the MATH table, we also see different behaviour after the `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 \LaTeX ; we can just map ' to `\prime` and not worry about it.

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

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

```

1462 \muskip_new:N \g_um_primekern_muskip
1463 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1464 \num_new:N \l_um_primecount_num
1465 \cs_new:Nn \um_nprimes:n {
1466   ^{
1467     \primesingle
1468     \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1469   }
1470 }
1471 \cs_new:Nn \um_nprimes_select:n {
1472   \prg_case_int:nnn {#1}{

```

```

1473 {1} { ^{\primesingle} }
1474 {2} {
1475 \um_glyph_if_exist:nTF {"2033} { ^{\primedouble} } {\um_nprimes:n {#1}}
1476 }
1477 {3} {
1478 \um_glyph_if_exist:nTF {"2034} { ^{\primetripel} } {\um_nprimes:n {#1}}
1479 }
1480 {4} {
1481 \um_glyph_if_exist:nTF {"2057} { ^{\primequadruple} } {\um_nprimes:n {#1}}
1482 }
1483 }{
1484 \um_nprimes:n {#1}
1485 }
1486 }

```

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.

```

1487 \cs_new:Nn \um_scanprime: {
1488 \num_zero:N \l_um_primecount_num
1489 \um_scanprime_collect:
1490 }
1491 \cs_new:Nn \um_scanprime_collect: {
1492 \num_incr:N \l_um_primecount_num
1493 \peek_meaning_remove:NTF ' {
1494 \um_scanprime_collect:
1495 }{
1496 \peek_meaning_remove:NTF \um_scanprime: {
1497 \um_scanprime_collect:
1498 }{
1499 \peek_meaning_remove:NTF ^^^2032 {
1500 \um_scanprime_collect:
1501 }{
1502 \um_nprimes_select:n {\l_um_primecount_num}
1503 }
1504 }
1505 }
1506 }
1507 \cs_set_eq:NN \prime \um_scanprime:
1508 \group_begin:
1509 \char_make_active:N \'
1510 \char_make_active:n {"2032}
1511 \cs_gset_eq:NN ' \um_scanprime:
1512 \cs_gset_eq:NN ^^^2032 \um_scanprime:
1513 \group_end:

```

8.0.2 Unicode radicals

Undo the damage made to `\sqrt`:

```
1514 \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`.

```
1515 \def\r@@t#1#2{
1516   \setbox\z@\hbox{$\math #1\sqrtsign{#2}$}
1517   \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1518   \raise \dimexpr(
1519     \um@fontdimen@percent{65}{\um@font}\ht\z@-
1520     \um@fontdimen@percent{65}{\um@font}\dp\z@
1521   )\relax
1522   \copy \rootbox
1523   \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1524   \box \z@
1525 }
```

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:

```
1526 \prop_new:N \g_um_supers_prop
1527 \prop_new:N \g_um_subs_prop
1528 \cs_generate_variant:Nn \prop_gput:Nnn {NxN}
1529 \cs_generate_variant:Nn \prop_get:NnN {cxN}
1530 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}
1531
1532 \group_begin:
1533
1534 % Populate a property list with superscript characters; their mean-
1535 % ing as their key,
1536 % for reasons that will become apparent soon, and their replace-
1537 % ment as each key's value.
1538 % Then make the superscript active and bind it to the scanning function.
```



```

1537 %
1538 % \cs{scantokens} makes this process much simpler since we can acti-
vate the char
1539 % and assign its meaning in one step.
1540 \cs_set:Nn \um_setup_active_superscript:nn {
1541   \prop_gput:Nxn \g_um_supers_prop   {\meaning #1} {#2}
1542   \char_make_active:n {\`#1}
1543   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1544   \scantokens{
1545     \cs_gset:Npn #1 {
1546       \tl_set:Nn \l_um_ss_chain_tl {#2}
1547       \cs_set_eq:NN \um_sub_or_super:n \sp
1548       \tl_set:Nn \l_um_tmpa_tl {supers}
1549       \um_scan_sscript:
1550     }
1551   }
1552 }
1553
1554 \um_setup_active_superscript:nn {^^^2070} {0}
1555 \um_setup_active_superscript:nn {^^^00b9} {1}
1556 \um_setup_active_superscript:nn {^^^00b2} {2}
1557 \um_setup_active_superscript:nn {^^^00b3} {3}
1558 \um_setup_active_superscript:nn {^^^2074} {4}
1559 \um_setup_active_superscript:nn {^^^2075} {5}
1560 \um_setup_active_superscript:nn {^^^2076} {6}
1561 \um_setup_active_superscript:nn {^^^2077} {7}
1562 \um_setup_active_superscript:nn {^^^2078} {8}
1563 \um_setup_active_superscript:nn {^^^2079} {9}
1564 \um_setup_active_superscript:nn {^^^207a} {+}
1565 \um_setup_active_superscript:nn {^^^207b} {-}
1566 \um_setup_active_superscript:nn {^^^207c} {=}
1567 \um_setup_active_superscript:nn {^^^207d} {(}
1568 \um_setup_active_superscript:nn {^^^207e} {)}
1569 \um_setup_active_superscript:nn {^^^2071} {i}
1570 \um_setup_active_superscript:nn {^^^207f} {n}
1571
1572 % Ditto above.
1573 \cs_set:Nn \um_setup_active_subscript:nn {
1574   \prop_gput:Nxn \g_um_subs_prop   {\meaning #1} {#2}
1575   \char_make_active:n {\`#1}
1576   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1577   \scantokens{
1578     \cs_gset:Npn #1 {
1579       \tl_set:Nn \l_um_ss_chain_tl {#2}
1580       \cs_set_eq:NN \um_sub_or_super:n \sb
1581       \tl_set:Nn \l_um_tmpa_tl {subs}

```

```

1582         \um_scan_sscript:
1583     }
1584 }
1585 }
1586
1587 \um_setup_active_subscript:nn {^^^2080} {0}
1588 \um_setup_active_subscript:nn {^^^2081} {1}
1589 \um_setup_active_subscript:nn {^^^2082} {2}
1590 \um_setup_active_subscript:nn {^^^2083} {3}
1591 \um_setup_active_subscript:nn {^^^2084} {4}
1592 \um_setup_active_subscript:nn {^^^2085} {5}
1593 \um_setup_active_subscript:nn {^^^2086} {6}
1594 \um_setup_active_subscript:nn {^^^2087} {7}
1595 \um_setup_active_subscript:nn {^^^2088} {8}
1596 \um_setup_active_subscript:nn {^^^2089} {9}
1597 \um_setup_active_subscript:nn {^^^208a} {+}
1598 \um_setup_active_subscript:nn {^^^208b} {-}
1599 \um_setup_active_subscript:nn {^^^208c} {=}
1600 \um_setup_active_subscript:nn {^^^208d} {(}
1601 \um_setup_active_subscript:nn {^^^208e} {)}
1602 \um_setup_active_subscript:nn {^^^2090} {a}
1603 \um_setup_active_subscript:nn {^^^2091} {e}
1604 \um_setup_active_subscript:nn {^^^1d62} {i}
1605 \um_setup_active_subscript:nn {^^^2092} {o}
1606 \um_setup_active_subscript:nn {^^^1d63} {r}
1607 \um_setup_active_subscript:nn {^^^1d64} {u}
1608 \um_setup_active_subscript:nn {^^^1d65} {v}
1609 \um_setup_active_subscript:nn {^^^2093} {x}
1610 \um_setup_active_subscript:nn {^^^1d66} {\beta}
1611 \um_setup_active_subscript:nn {^^^1d67} {\gamma}
1612 \um_setup_active_subscript:nn {^^^1d68} {\rho}
1613 \um_setup_active_subscript:nn {^^^1d69} {\phi}
1614 \um_setup_active_subscript:nn {^^^1d6a} {\chi}
1615
1616 \group_end:
1617
1618 % The scanning command, evident in its purpose:
1619 \cs_new:Nn \um_scan_sscript: {
1620     \um_scan_sscript:TF {
1621         \um_scan_sscript:
1622     }{
1623         \um_sub_or_super:n {\l_um_ss_chain_tl}
1624     }
1625 }
1626
1627 % The main theme here is stolen from the source to the vari-

```

```

        ous \cs{peek_} functions.
1628 % Consider this function as simply boilerplate:
1629 \cs_new:Nn \um_scan_sscript:TF {
1630   \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
1631   \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
1632   \tl_set:Nx \l_peek_false_tl { \exp_not:n{ \group_align_safe_end: #2 } }
1633   \group_align_safe_begin:
1634   \peek_after:NN \um_peek_execute_branches_ss:
1635 }
1636
1637 % We do not skip spaces when scanning ahead, and we explicitly wish to
1638 % bail out on encountering a space or a brace.
1639 \cs_new:Npn \um_peek_execute_branches_ss: {
1640   \bool_if:nTF {
1641     \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
1642     \token_if_eq_catcode_p:NN \l_peek_token \c_group_end_token ||
1643     \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
1644   }
1645   { \l_peek_false_tl }
1646   { \um_peek_execute_branches_ss_aux: }
1647 }
1648
1649 % This is the actual comparison code.
1650 % Because the peeking has already tokenised the next token,
1651 % it's too late to extract its charcode directly. Instead,
1652 % we look at its meaning, which remains a `character' even
1653 % though it is itself math-active. If the character is ever
1654 % made fully active, this will break our assumptions!
1655 %
1656 % If the char's meaning exists as a property list key, we
1657 % build up a chain of sub-/superscripts and iterate. (If not, exit and
1658 % typeset what we've already collected.)
1659 \cs_new:Nn \um_peek_execute_branches_ss_aux: {
1660   \prop_if_in:cxTF
1661     {g_um_\l_um_tmpa_tl _prop}
1662     {\meaning\l_peek_token}
1663   {
1664     \prop_get:cxN
1665       {g_um_\l_um_tmpa_tl _prop}
1666       {\meaning\l_peek_token}
1667     \l_um_tmpb_tl
1668     \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmpb_tl
1669     \l_peek_true_tl
1670   }
1671   {\l_peek_false_tl}
1672 }

```

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`:

```
1673 \def\operator@font{\um_setup_mathup:}
1674 \def\to{\rightarrow}
1675 \def\vec{\overrightarrow}
1676 \def\le{\leq}
1677 \def\ge{\geq}
1678 \def\neq{\neq}
```

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

```
1679 \@ifpackageloaded{amsmath}{
1680   % define their own colon, perhaps I should just steal it.
1681 }{
1682   \cs_set_protected:Npn \colon {
1683     \bool_if:NTF \g_um_literal_colon_bool {:} { \mathpunct{:} }
1684   }
1685 }
```

`\mathcal`

```
1686 \def\mathcal{\mathscr}
```

`\mathrm`

```
1687 \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:

```
1688 \ifpackageloaded{amsmath}{
1689   \tl_remove_in:Nn \@begindocumenthook {
1690     \mathchardef\std@minus\mathcode`\-\relax
1691     \mathchardef\std@equal\mathcode`\=\relax
1692   }
1693 }
```

- 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:

```
1694 \ifpackageloaded{amsopn}{
```

```

1695 \cs_set:Npn \newmcodes@ {
1696   \mathcode`\'39
1697   \mathcode`\'*42
1698   \mathcode`\'."613A%
1699   % \ifnum\mathcode`\'=-45 \else
1700   %   \mathchardef\std@minus\mathcode`\'-\relax
1701   % \fi
1702   \mathcode`\'-45
1703   \mathcode`\'/47
1704   \mathcode`\':"603A\relax
1705 }
1706 {}{}

```

Octothorpe is an odd one:

```

1707 \AtBeginDocument{
1708   \def\#{\mode_if_math:TF{\mathoctothorpe}{\char`\'#}}
1709   \def\widehat{\hat}
1710   \def\widetilde{\tilde}
1711 }

```

`\digamma` I might end up just changing these in the table.

```

\Digamma
1712 \def\digamma{\updigamma}
1713 \def\Digamma{\upDigamma}

```

Overriding amsmath definitions:

```

1714 \AtBeginDocument{
1715   \def@cdots{\mathinner{\cdots}}
1716 }

```

Interaction with beamer:

```

1717 \@ifclassloaded{beamer}{
1718   \ifbeamer@suppressreplacements\else
1719     \PackageWarningNoLine{unicode-math}{
1720       Disabling~ beamer's~ math~ setup.^J
1721       Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1722     }
1723     \beamer@suppressreplacementstrue
1724   \fi
1725 }{}

```

The end.

```

1726 \ExplSyntaxOff

```

File II

STIX table data extraction

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

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

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

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

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

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

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

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

Print the actual entry corresponding to the unicode character:

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

Now replace the STIX class abbreviations with their T_EX macro names.

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

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

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

Fixing up a couple of things in the STIX table.

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

A Documenting maths support in the NFSS

A.1 Overview

In the following, *<NFSS decl.>* stands for something like `{T1}{lmr}{m}{n}`.

Maths symbol fonts Fonts for symbols: α , \leq , \rightarrow

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

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

Maths alphabet fonts Fonts for *ABC-xyz*, *ℳℳℳ-ℳℳℳ*, etc.

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

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

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

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

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

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

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

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

Delimiters and radicals use wrappers around \TeX 's `\delimiter`/`\radical` primitives, which are re-designed in \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	SCRIPTSCRIPTPERCENTSCALE-DOWN	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	DELIMITEDSUBFORMULAMIN-HEIGHT	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height \times 1.5.
13	DISPLAYOPERATORMINHEIGHT	Minimum height of n-ary operators (such as integral and summation) for formulas in display mode.
14	MATHLEADING	White space to be left between math formulas to ensure proper line spacing. For example, for applications that treat line gap as a part of line ascender, formulas with ink going above (os2.sTypoAscender + os2.sTypoLineGap – MathLeading) or with ink going below os2.sTypoDescender will result in increasing line height.
15	AXISHEIGHT	Axis height of the font.
16	ACCENTBASEHEIGHT	Maximum (ink) height of accent base that does not require raising the accents. Suggested: x-height of the font (os2.sxHeight) plus any possible overshots.
17	FLATTENEDACCENTBASE-HEIGHT	Maximum (ink) height of accent base that does not require flattening the accents. Suggested: cap height of the font (os2.sCapHeight).
18	SUBSCRIPTSHIFTDOWN	The standard shift down applied to subscript elements. Positive for moving in the downward direction. Suggested: os2.ySubscriptYOffset.
19	SUBSCRIPTTOPMAX	Maximum allowed height of the (ink) top of subscripts that does not require moving subscripts further down. Suggested: $/5$ x-height.

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

\fontdimen	Dimension name	Description
31	LOWERLIMITBASELINEDROP-MIN	Minimum distance between baseline of the lower limit and (ink) bottom of the base operator.
32	STACKTOPSHIFTUP	Standard shift up applied to the top element of a stack.
33	STACKTOPDISPLAYSTYLESHIFTUP	Standard shift up applied to the top element of a stack in display style.
34	STACKBOTTOMSHIFTDOWN	Standard shift down applied to the bottom element of a stack. Positive for moving in the downward direction.
35	STACKBOTTOMDISPLAYSTYLESHIFTDOWN	Standard shift down applied to the bottom element of a stack in display style. Positive for moving in the downward direction.
36	STACKGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element. Suggested: 3×default rule thickness.
37	STACKDISPLAYSTYLEGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element in display style. Suggested: 7×default rule thickness.
38	STRETCHSTACKTOPSHIFTUP	Standard shift up applied to the top element of the stretch stack.
39	STRETCHSTACKBOTTOMSHIFTDOWN	Standard shift down applied to the bottom element of the stretch stack. Positive for moving in the downward direction.
40	STRETCHSTACKGAPABOVEMIN	Minimum gap between the ink of the stretched element, and the (ink) bottom of the element above. Suggested: UpperLimitGapMin
41	STRETCHSTACKGAPBELOWMIN	Minimum gap between the ink of the stretched element, and the (ink) top of the element below. Suggested: LowerLimitGapMin.
42	FRACTIONNUMERATORSHIFTUP	Standard shift up applied to the numerator.
43	FRACTIONNUMERATORDISPLAYSTYLESHIFTUP	Standard shift up applied to the numerator in display style. Suggested: StackTopDisplayStyleShiftUp.

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

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

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
\#	1453, 1708
\'	1509, 1696
*	495, 1457, 1697
\-	494, 1690, 1699, 1700, 1702
\.	1698
\/	544, 1703
\:	497, 1704
\::	781
\::f	781
\::n	781
\<	548
\=	1691
\>	549
\@DeclareMathDelimiter	340
\@DeclareMathSizes	305
\@backslashchar	971
\@beginofdocumenthook	1689
\@cclvi	365
\@cdots	1715
\@elt	652–656, 659, 663, 665
\@empty	416, 417, 454, 581, 697, 704, 719, 740, 745
\@ifclassloaded	1717
\@ifnextchar	1514
\@ifpackageloaded	1679, 1688, 1694
\@ii	703, 704, 706, 708, 711, 716
\@input	471, 1454
\@marker	716, 735
\@nil	374, 525, 716, 729–732, 1433, 1438, 1451
\@preamblecmds	349
\@sqrt	1514
\@tempa	133, 180, 212, 238, 256, 274, 282, 289, 427, 443, 695, 707, 731, 733, 745
\@tempb	133, 134, 180, 181, 212, 213, 238, 239, 256, 257, 274, 275, 282, 283, 289, 290, 695, 696, 734, 735, 740
\@tempswafalse	705
\@tempswatru	709, 712, 737, 742, 747, 752
\@um@bfliteraltrue	175, 200
\@um@bfupGreekfalse	139, 182
\@um@bfupGreektrue	152, 165, 188, 194
\@um@bfupLatinfalse	142, 184
\@um@bfupLatintrue	155, 168, 190, 196
\@um@bfupgreekfalse	140, 153, 183, 189
\@um@bfupgreektrue	166, 195
\@um@bfuplatinfalse	143, 185
\@um@bfuplatintrue	156, 169, 191, 197
\@um@fontspec@featuretrue	418
\@um@literaltrue	174
\@um@ot@math@true	446
\@um@upGreekfalse	135
\@um@upGreektrue	148, 161
\@um@upLatinfalse	137, 150
\@um@upLatintrue	163
\@um@upNablafalse	144, 242
\@um@upNablatrue	157, 170, 240
\@um@upgreekfalse	136, 149
\@um@upgreektrue	162
\@um@uplatinfalse	138, 151, 164
\@um@uppartialfalse	141, 154, 186, 192, 260
\@um@uppartialtrue	167, 198, 258
\@xDeclareMathDelimiter	341
\@xxDeclareMathDelimiter	339
\\	10–13, 17, 23–33, 1457, 1459
\^	33, 1432, 1442
\	1458
Numbers	
\0	23
_	17–20, 23–32
A	
\A	24
\a	25
\addnolimits	<u>658</u>

\addtoversion	312	\clist_map_break:	962
\alloc@	365	\clist_map_inline:Nn	638
\Alpha	808	\clist_map_inline:nn	
\alpha	833	530, 633, 640, 958, 965
\alpha@elt	316	\clist_map_variable:NNn	703
\alpha@list	315	\clist_map_variable:nNn ..	760, 783, 789
\AtBeginDocument	806, 1707, 1714	\colon	1682
\awint	654	\copy	1522
B		\cs	1538, 1627
\B	1067	\cs_generate_variant:Nn ...	1528–1530
\beamer@suppressreplacementstrue	1723	\cs_gset:cpn	379, 389
\begingroup	371, 662, 1431, 1449	\cs_gset:Npn ..	392, 397, 1433, 1545, 1578
\Beta	809	\cs_gset:Npx	402
\beta	834, 1610	\cs_gset_eq:NN	1511, 1512
\bgroup	995	\cs_if_exist:cF	990
\bool_if:NF	496	\cs_new:Nn	479, 493, 524, 529,
\bool_if:NTF	222, 838, 857, 863,	534, 537, 543, 574, 577, 782, 788,
.....	868, 1096, 1104, 1110, 1304, 1312,	797, 800, 803, 938, 957, 989, 1465,
.....	1318, 1326, 1336, 1344, 1364, 1683	1471, 1487, 1491, 1619, 1629, 1659
\bool_if:nTF	1640	\cs_new:Npn	1004,
\bool_new:N	22, 209, 210, 281	1007, 1011, 1016, 1026, 1029,
\bool_set_false:N	1033, 1038, 1048, 1052, 1062,
.....	145, 146, 172, 177, 214, 276, 286	1065, 1076, 1083, 1091, 1095,
\bool_set_true:N	1103, 1117, 1120, 1123, 1127,
.....	158, 159, 171, 176, 216, 218, 278, 284	1130, 1133, 1136, 1140, 1143,
\box	1524	1146, 1150, 1153, 1161, 1175,
C		1194, 1233, 1236, 1239, 1243,
\C	1054, 1085	1248, 1259, 1262, 1265, 1269,
\c_group_begin_token	1641	1275, 1286, 1289, 1293, 1296,
\c_group_end_token	1642	1300, 1303, 1311, 1325, 1343,
\c_peek_true_remove_next_tl ...	1631	1381, 1384, 1387, 1391, 1396,
\c_space_token	1643	1406, 1409, 1412, 1416, 1421, 1639
\cdots	1715	\cs_set:cpn	991
\cdp@elt	303	\cs_set:Nn	203, 221, 245,
\cdp@list	302	263, 411, 485, 488, 580, 632, 637,
\char	1708	759, 769, 772, 775, 778, 873, 884,
\char_make_active:N	1509	894, 901, 910, 919, 983, 1540, 1573
\char_make_active:n	372, 1510, 1542, 1575	\cs_set:Npn	781, 1695
\char_make_escape:N	1458	\cs_set_eq:cN	984
\char_make_math_superscript:N ..	1442	\cs_set_eq:NN	
\char_make_other:N ..	1432, 1453, 1459	457–460, 464–467, 1507, 1547, 1580
\chardef	365	\cs_set_protected:cpn	994
\Chi	830	\cs_set_protected:Npn	1682
\chi	859, 1614	\cs_to_str:N	374, 379, 389, 647
\cirfnint	654	\csname	368, 374, 377, 380,
		412, 419, 539, 575, 578, 959, 966, 998

D	
<code>\DeclareDocumentCommand</code>	414
<code>\DeclareMathAccent</code>	333
<code>\DeclareMathAlphabet</code>	329
<code>\DeclareMathDelimiter</code>	338
<code>\DeclareMathRadical</code>	344
<code>\DeclareMathSizes</code>	304
<code>\DeclareMathSymbol</code>	335
<code>\DeclareMathVersion</code>	309, 421
<code>\DeclareRobustCommand</code>	1514
<code>\DeclareSymbolFont</code>	322, 469
<code>\DeclareSymbolFontAlphabet</code>	346
<code>\DeclareSymbolFontAlphabet@</code>	347
<code>\def</code>	23–131, 351, 354, 365, 367, 369, 420, 651, 659, 661, 663, 670, 672, 729, 731–734, 808–837, 840–856, 859–862, 865–867, 870, 871, 1444, 1450, 1515, 1673–1678, 1686, 1687, 1708–1710, 1712, 1713, 1715
<code>\define@choicekey</code>	132, 180, 211, 238, 256, 274, 282, 289, 695
<code>\define@cmdkey</code>	691–694
<code>\define@key</code>	674
<code>\define@mathalphabet</code>	310
<code>\define@mathgroup</code>	311
<code>\Delta</code>	811
<code>\delta</code>	836
<code>\Digamma</code>	<u>1712</u>
<code>\digamma</code>	<u>1712</u>
<code>\dimexpr</code>	352, 445, 1518
<code>\do</code>	349
<code>\dorestore@version</code>	319
<code>\dp</code>	1520
E	
<code>\E</code>	1068
<code>\e</code>	1078
<code>\edef</code>	427, 730, 731
<code>\egroup</code>	992, 997
<code>\else</code>	204–207, 250, 268, 357, 360, 386, 390, 395, 400, 403, 447, 461, 504, 517, 585, 593, 596, 601, 607, 619, 629, 664, 677, 710, 715, 721, 739, 744, 749, 897, 905, 914, 928, 1157, 1166, 1170, 1184, 1189, 1213, 1222, 1699, 1718
<code>\else:</code>	987
<code>\encodingdefault</code>	470
<code>\endcsname</code>	368, 374, 377, 380, 412, 419, 539, 575, 578, 959, 966, 998
<code>\endgroup</code>	375, 668, 1448, 1455
<code>\Epsilon</code>	812
<code>\epsilon</code>	837
<code>\Eta</code>	814
<code>\eta</code>	841
<code>\etex_iffontchar:D</code>	987
<code>\ExecuteOptionsX</code>	298
<code>\exp_args:Nnff</code>	781, 784, 791
<code>\exp_args:No</code>	634, 641
<code>\exp_not:n</code>	1630, 1632
<code>\expandafter</code>	368, 376, 381, 383, 387, 659, 706, 708, 711, 716, 730, 731, 735, 997, 998
<code>\ExplSyntaxOff</code>	1726
<code>\ExplSyntaxOn</code>	6, 1441
F	
<code>\F</code>	1069
<code>\f@size</code>	419
<code>\fi</code>	178, 201, 204–207, 219, 243, 254, 261, 272, 279, 287, 296, 362, 363, 384, 405–409, 453, 468, 507, 522, 590, 598, 603, 610, 627, 628, 630, 643, 666, 682, 698, 713, 714, 717, 723, 725, 726, 738, 743, 748, 753–757, 899, 908, 917, 936, 1159, 1172, 1173, 1191, 1192, 1229, 1231, 1701, 1724
<code>\fi:</code>	987
<code>\fint</code>	654
<code>\font</code>	444
<code>\fontdimen</code>	352, 445, 1517, 1523
<code>\fontname</code>	974
G	
<code>\g</code>	1079
<code>\g@addto@macro</code>	720, 722
<code>\g_um_literal_colon_bool</code>	281, 284, 286, 496, 1683
<code>\g_um_math_alphabet_name_Greek_tl</code>	981
<code>\g_um_math_alphabet_name_greek_tl</code>	980

<code>\l_um_incr_num</code>	761, 763, 765
<code>\l_um_input_num</code>	760, 763, 783, 785, 789, 792
<code>\l_um_primecount_num</code>	1464, 1488, 1492, 1502
<code>\l_um_script_features_tl</code> ...	422, 433
<code>\l_um_script_font_tl</code>	424, 432
<code>\l_um_ss_chain_tl</code>	1546, 1579, 1623, 1668
<code>\l_um_sscript_features_tl</code> ..	423, 437
<code>\l_um_sscript_font_tl</code>	425, 436
<code>\l_um_tmppa_tl</code> ...	1548, 1581, 1661, 1665
<code>\l_um_tmppb_tl</code>	1667, 1668
<code>\Lambda</code>	818
<code>\lambda</code>	845
<code>\lccode</code>	1457
<code>\le</code>	1676
<code>\left</code>	671
<code>\left@primitive</code>	671, 672
<code>\leq</code>	1676
<code>\let</code>	366, 415–417, 671, 697, 1435
<code>\lowercase</code>	1434, 1439
<code>\lowint</code>	656
M	
<code>\M</code>	1073
<code>\m@th</code>	1516
<code>\mathaccent</code>	401
<code>\mathalpha</code>	648, 764
<code>\mathbb</code>	1049, 1050, 1053–1060, 1063
<code>\mathbf</code>	1151, 1155, 1156, 1158, 1163–1165, 1167, 1169, 1171, 1176, 1178–1183, 1185, 1186, 1188, 1190, 1195, 1197–1212, 1214, 1216–1221, 1223–1228, 1230
<code>\mathbffrac</code>	1287, 1290, 1291
<code>\mathbfrit</code>	1234, 1237, 1240, 1241, 1244–1246, 1249–1257
<code>\mathbfscr</code>	1294, 1297, 1298
<code>\mathbfssf</code>	1301, 1305, 1306, 1308, 1313–1315, 1317, 1319, 1321, 1327–1332, 1334, 1335, 1337, 1339, 1345–1360, 1362, 1363, 1365–1370, 1372–1377
<code>\mathbfsfrit</code>	1407, 1410, 1413, 1414, 1417–1419, 1422–1429
<code>\mathbfsfup</code>	1382, 1385, 1388, 1389, 1392–1394, 1397–1404
<code>\mathbfup</code>	1260, 1263, 1266, 1267, 1270–1273, 1276–1284
<code>\mathbin</code>	494, 495
<code>\mathcal</code>	1686
<code>\mathchar@type</code>	345, 378, 392, 394, 397, 399, 402, 404, 412, 538
<code>\mathchardef</code>	1690, 1691, 1700
<code>\mathclose</code>	396
<code>\mathcode</code>	373, 1690, 1691, 1696–1700, 1702–1704
<code>\mathfrac</code>	1084–1089, 1092, 1093
<code>\mathgroup</code>	365
<code>\mathinner</code>	1715
<code>\mathit</code>	1027, 1030, 1031, 1034–1036, 1039–1046
<code>\mathoctothorpe</code>	1708
<code>\mathop</code>	370
<code>\mathopen</code>	391, 672
<code>\mathord</code>	500–503, 505, 506, 509–516, 518–521, 535
<code>\mathpunct</code>	1683
<code>\mathrel</code>	497
<code>\mathrm</code>	1687
<code>\mathscr</code> ...	1066–1074, 1077–1081, 1686
<code>\mathsf</code>	1097, 1098, 1100, 1105–1107, 1109, 1111, 1113, 1118
<code>\mathsfrit</code>	1131, 1134, 1137, 1138
<code>\mathsfup</code>	1121, 1124, 1125, 1128
<code>\mathhtt</code>	1141, 1144, 1147, 1148
<code>\mathup</code>	1005, 1008, 1009, 1012–1014, 1017–1024, 1687
<code>\mddefault</code>	470
<code>\meaning</code>	1541, 1574, 1662, 1666
<code>\mitAlpha</code>	808
<code>\mitalpha</code>	833
<code>\mitBeta</code>	809
<code>\mitbeta</code>	834
<code>\mitChi</code>	830
<code>\mitchi</code>	859
<code>\mitDelta</code>	811
<code>\mitdelta</code>	836
<code>\mitEpsilon</code>	812
<code>\mitepsilon</code>	838, 863
<code>\mitEta</code>	814
<code>\miteta</code>	841
<code>\mitGamma</code>	810

\backslash overrightarrow	1675	\backslash raise	1518
P		\backslash relax	134, 181, 213, 239, 257, 275, 283, 290, 352, 370, 373, 378, 389, 391–394, 396–399, 401, 402, 404, 412, 415, 444, 445, 639, 696, 708, 711, 735, 736, 741, 746, 750, 751, 763, 765, 785, 792, 793, 1521, 1690, 1691, 1700, 1704
\backslash P	1057	\backslash removenolimits	<u>661</u>
\backslash PackageError	678	\backslash RequirePackage	3–5
\backslash PackageInfo	456	\backslash restore@mathversion	317
\backslash PackageWarningNoLine ..	448, 969, 1719	\backslash Rho	824
\backslash peek_after:NN	1634	\backslash rho	851, 1612
\backslash peek_meaning_remove:NTF	1493, 1496, 1499	\backslash rightarrow	1674
\backslash Phi	829	\backslash rootbox	1522
\backslash phi	856, 1613	\backslash rppolint	654
\backslash Pi	823	S	
\backslash pi	850	\backslash sb	1580
\backslash pointint	655	\backslash scan_stop: ...	540, 541, 987, 1543, 1576
\backslash prg_case_int:nnn	1472	\backslash scantokens	1435, 1544, 1577
\backslash prg_do_nothing:	984	\backslash scpolint	655
\backslash prg_new_conditional:Nnn	986	\backslash scriptscriptstyle	358
\backslash prg_replicate:nn	1468	\backslash scriptstyle	355
\backslash prg_return_false:	987	\backslash set@@mathdelimiter	343
\backslash prg_return_true:	987	\backslash set@mathaccent	334
\backslash prg_stepwise_variable:nnnNn	761, 790	\backslash set@mathchar	336
\backslash prime	1507	\backslash set@mathdelimiter	342
\backslash primedouble	1475	\backslash set@mathsymbol	337
\backslash primequadruple	1481	\backslash setbox	1516
\backslash primesingle	535, 1467, 1468, 1473	\backslash setkeys	426
\backslash primetriples	1478	\backslash SetMathAlphabet	331
\backslash process@table	320	\backslash SetMathAlphabet@	332
\backslash ProcessOptionsX	299	\backslash setmathfont	<u>414</u> , 681
\backslash prop_get:cxN	1664	\backslash SetSymbolFont	326
\backslash prop_get:NnN	1529	\backslash SetSymbolFont@	327
\backslash prop_gput:Nnn	1528	\backslash sf@size	431, 435
\backslash prop_gput:Nxn	1541, 1574	\backslash Sigma	826
\backslash prop_if_in:cxTF	1660	\backslash sigma	853
\backslash prop_if_in:NnTF	1530	\backslash sp	1547
\backslash prop_new:N	1526, 1527	\backslash space	969, 998
\backslash protect	681	\backslash sqint	655
\backslash ProvidesPackage	1	\backslash sqrt	670, 1514
\backslash Psi	831	\backslash sqrtsign	1514, 1516
\backslash psi	860	\backslash std@equal	1691
Q		\backslash std@minus	1690, 1700
\backslash Q	1058	\backslash stepcounter	462
R			
\backslash R	1059, 1074, 1088		
\backslash r@@t	<u>1515</u>		

<code>\string</code>	377, 380, 730, 731	<code>\um@scaled@apply</code>	<u>354</u> , 1517, 1523
<code>\strip@pt</code>	352	<code>\um@scanactivedef</code>	374, <u>1431</u>
<code>\sumint</code>	653	<code>\um@scancharlet</code>	<u>1431</u> , 1451
T			
<code>\Tau</code>	827	<code>\um@set@mathsymbol</code>	368, <u>369</u>
<code>\tau</code>	854	<code>\um@usv@bbLatin</code>	33, 1053
<code>\tf@size</code>	430, 431	<code>\um@usv@bblatin</code>	34, 1049
<code>\Theta</code>	815	<code>\um@usv@bbnum</code>	32, 1063
<code>\theta</code>	842	<code>\um@usv@bfDigamma</code>	95, 1176, 1273
<code>\theum@fam</code>	463	<code>\um@usv@bfdigamma</code>	102, 1195, 1284
<code>\thinmuskip</code>	1463	<code>\um@usv@bffrakLatin</code>	66, 1287
<code>\tilde</code>	1710	<code>\um@usv@bffraklatin</code>	67, 1290
<code>\tl_map_inline:nn</code>	300	<code>\um@usv@bfGreek</code>	60
<code>\tl_put_right:cn</code>	647	<code>\um@usv@bfgreek</code>	61
<code>\tl_put_right:NV</code>	1668	<code>\um@usv@bfitGreek</code> 64, 204, 605, 608, 891, 1179, 1244
<code>\tl_remove_in:Nn</code>	349, 1689	<code>\um@usv@bfitgreek</code> 65, 205, 612, 620, 892, 1198, 1250
<code>\tl_rescan:nn</code>	1440	<code>\um@usv@bfith</code>	112, 1165, 1171
<code>\tl_set:Nn</code>	223–228, 230–235, 247–249, 251–253, 265–267, 269–271, 291, 293, 295, 422–425, 455, 978–982, 1546, 1548, 1579, 1581	<code>\um@usv@bfitLatin</code>	62, 206, 595, 597, 887, 1156, 1237, 1249
<code>\tl_set:Nx</code>	204–207, 463, 1630, 1632	<code>\um@usv@bfitlatin</code> 63, 207, 600, 602, 888, 1164, 1240
<code>\tl_set_eq:NN</code>	1631	<code>\um@usv@bfitNabla</code> 123, 252, 510, 518, 1183, 1246
<code>\tl_use:c</code>	972	<code>\um@usv@bfitnum</code>	53
<code>\to</code>	1674	<code>\um@usv@bfitpartial</code> 129, 270, 514, 520, 1206, 1251
<code>\token_if_eq_catcode_p:NN</code>	1641, 1642	<code>\um@usv@bfitvarepsilon</code> 114, 613, 621, 1207, 1223, 1252
<code>\token_if_eq_meaning_p:NN</code>	1643	<code>\um@usv@bfitvarkappa</code> 116, 615, 623, 1209, 1225, 1254
U			
<code>\um@addto@mathmap</code>	634, 641, <u>646</u>	<code>\um@usv@bfitvarphi</code> 117, 616, 624, 1210, 1226, 1255
<code>\um@backslash</code>	707, 730	<code>\um@usv@bfitvarpi</code> 119, 618, 626, 1212, 1228, 1257
<code>\um@char@num@range</code>	417, 638, 719, 720, 722	<code>\um@usv@bfitvarrho</code> 118, 617, 625, 1211, 1227, 1256
<code>\um@char@range</code>	416, 454, 581, 697, 700, 703	<code>\um@usv@bfitvarTheta</code> 113, 606, 609, 1181, 1190, 1245
<code>\um@firstchar</code>	706, 731	<code>\um@usv@bfitvartheta</code> 115, 614, 622, 1208, 1224, 1253
<code>\um@firsttof</code>	729–731	<code>\um@usv@bfLatin</code>	55
<code>\um@font</code>	356, 359, 444, 445, 974, 987, 1517, 1519, 1520, 1523	<code>\um@usv@bflatin</code>	57
<code>\um@fontdimen@percent</code> 351, 356, 359, 1519, 1520	<code>\um@usv@bfNabla</code> 122, 248, 509, 518, 1182, 1272
<code>\um@mathsymbol</code>	<u>367</u> , 486	<code>\um@usv@bfnum</code>	51, 1151, 1234, 1260
<code>\um@mversion</code>	420, 421		
<code>\um@nolimits</code>	381, <u>651</u> , 659, 667		
<code>\um@parse@range</code>	716, <u>732</u>		
<code>\um@parse@term</code>	489, 525, <u>702</u>		
<code>\um@radicals</code>	387, <u>670</u>		
<code>\um@resolve@greek</code>	<u>806</u>		

$\backslash\mathrm{um@usv@bfpartial}$	$\backslash\mathrm{um@usv@bfvarTheta}$
..... 128, 266, 513, 520, 1199, 1277 94, 606, 609, 1180, 1188, 1271
$\backslash\mathrm{um@usv@bfscrLatin}$	$\backslash\mathrm{um@usv@bfvartheta}$
..... 68, 1294 97, 614, 622, 1201, 1217, 1279
$\backslash\mathrm{um@usv@bfscrLatin}$	$\backslash\mathrm{um@usv@Digamma}$
..... 69, 1297 86, 1176, 1273
$\backslash\mathrm{um@usv@bfsfGreek}$	$\backslash\mathrm{um@usv@digamma}$
..... 79 93, 1195, 1284
$\backslash\mathrm{um@usv@bfsfgreek}$	$\backslash\mathrm{um@usv@frakLatin}$
..... 80 37, 1084
$\backslash\mathrm{um@usv@bfsfitGreek}$ 83, 234, 1328, 1417	$\backslash\mathrm{um@usv@fraklatin}$
$\backslash\mathrm{um@usv@bfsfitgreek}$ 84, 235, 1346, 1422 38, 1092
$\backslash\mathrm{um@usv@bfsfitLatin}$ 81, 232, 1306, 1410	$\backslash\mathrm{um@usv@itGreek}$
$\backslash\mathrm{um@usv@bfsfitlatin}$ 82, 233, 1314, 1413 30, 881,
$\backslash\mathrm{um@usv@bfsfitNabla}$ 912, 915, 1012, 1034, 1179, 1185,
..... 125, 253, 512, 519, 1332, 1419 1244, 1270, 1328, 1334, 1392, 1417
$\backslash\mathrm{um@usv@bfsfitnum}$	$\backslash\mathrm{um@usv@itgreek}$
..... 72 31,
$\backslash\mathrm{um@usv@bfsfitpartial}$ 921, 929, 1017, 1039, 1198, 1214,
..... 131, 271, 516, 521, 1354, 1423 1250, 1276, 1346, 1362, 1397, 1422
$\backslash\mathrm{um@usv@bfsfLatin}$	$\backslash\mathrm{um@usv@ith}$
..... 74 103, 877, 904, 907, 1009, 1031,
$\backslash\mathrm{um@usv@bfsflatin}$ 1050, 1081, 1093, 1107, 1111,
..... 76 1113, 1125, 1138, 1148, 1165,
$\backslash\mathrm{um@usv@bfsfNabla}$ 124, 249, 511, 519, 1331 1169, 1171, 1241, 1267, 1291,
$\backslash\mathrm{um@usv@bfsfnum}$... 70, 1301, 1382, 1407 1298, 1315, 1319, 1321, 1389, 1414
$\backslash\mathrm{um@usv@bfsfpartial}$	$\backslash\mathrm{um@usv@itLatin}$
..... 130, 267, 515, 521, 1347 28, 875, 896, 898, 1005,
$\backslash\mathrm{um@usv@bfsfupGreek}$ 77, 227, 1327, 1392 1027, 1053, 1066, 1084, 1098,
$\backslash\mathrm{um@usv@bfsfupgreek}$ 78, 228, 1345, 1397 1100, 1128, 1134, 1144, 1156,
$\backslash\mathrm{um@usv@bfsfupLatin}$ 73, 225, 1305, 1385 1158, 1237, 1249, 1263, 1287,
$\backslash\mathrm{um@usv@bfsfuplatin}$ 75, 226, 1313, 1388 1294, 1306, 1308, 1317, 1385, 1410
$\backslash\mathrm{um@usv@bfsfupnum}$	$\backslash\mathrm{um@usv@itlatin}$
..... 71 29,
$\backslash\mathrm{um@usv@bfupGreek}$ 876, 903, 906, 1008, 1030, 1049,
..... 58, 204, 605, 608, 889, 1178, 1270 1077, 1092, 1106, 1109, 1124,
$\backslash\mathrm{um@usv@bfupgreek}$ 1137, 1147, 1164, 1167, 1240,
..... 59, 205, 612, 620, 890, 1197, 1276 1266, 1290, 1297, 1314, 1388, 1413
$\backslash\mathrm{um@usv@bfuph}$	$\backslash\mathrm{um@usv@itNabla}$
..... 111, 1169 121, 251,
$\backslash\mathrm{um@usv@bfupLatin}$ 501, 505, 1013, 1036, 1183, 1186,
..... 54, 206, 595, 597, 885, 1155, 1263 1246, 1272, 1332, 1335, 1394, 1419
$\backslash\mathrm{um@usv@bfuplatin}$	$\backslash\mathrm{um@usv@itpartial}$
..... 56, 207, 600, 602, 886, 1163, 1266 127, 269,
$\backslash\mathrm{um@usv@bfupnum}$ 503, 506, 1018, 1040, 1206, 1230,
..... 52 1251, 1277, 1354, 1363, 1398, 1423
$\backslash\mathrm{um@usv@bfvarepsilon}$	$\backslash\mathrm{um@usv@itvarepsilon}$
..... 96, 613, 621, 1200, 1216, 1278 105, 922, 930, 1019,
$\backslash\mathrm{um@usv@bfvarkappa}$ 1041, 1207, 1216, 1223, 1252,
..... 98, 615, 623, 1202, 1218, 1280 1278, 1355, 1365, 1372, 1399, 1424
$\backslash\mathrm{um@usv@bfvarphi}$	$\backslash\mathrm{um@usv@itvarkappa}$
..... 99, 616, 624, 1203, 1219, 1281 107, 924, 932, 1021,
$\backslash\mathrm{um@usv@bfvarpi}$ 1043, 1209, 1218, 1225, 1254,
..... 101, 618, 626, 1205, 1221, 1283 1280, 1357, 1367, 1374, 1401, 1426
$\backslash\mathrm{um@usv@bfvarrho}$	$\backslash\mathrm{um@usv@itvarphi}$ 108, 925, 933, 1022,
..... 100, 617, 625, 1204, 1220, 1282	

1044, 1210, 1219, 1226, 1255, 1281, 1358, 1368, 1375, 1402, 1427	1027, 1053, 1066, 1084, 1097, 1100, 1128, 1134, 1144, 1155, 1158, 1237, 1249, 1263, 1287, 1294, 1305, 1308, 1317, 1385, 1410
\um@usv@itvarpi 110, 927, 935, 1024, 1046, 1212, 1221, 1228, 1257, 1283, 1360, 1370, 1377, 1404, 1429	\um@usv@uplatin 25, 878, 903, 906, 1008, 1030, 1049, 1077, 1092, 1105, 1109, 1124, 1137, 1147, 1163, 1167, 1240, 1266, 1290, 1297, 1313, 1388, 1413
\um@usv@itvarrho 109, 926, 934, 1023, 1045, 1211, 1220, 1227, 1256, 1282, 1359, 1369, 1376, 1403, 1428	\um@usv@varepsilon 87, 922, 930, 1019, 1041, 1200, 1216, 1223, 1252, 1278, 1348, 1365, 1372, 1399, 1424
\um@usv@itvarTheta 104, 916, 1014, 1035, 1181, 1188, 1190, 1245, 1271, 1337, 1339, 1393	\um@usv@varkappa 89, 924, 932, 1021, 1043, 1202, 1218, 1225, 1254, 1280, 1350, 1367, 1374, 1401, 1426
\um@usv@itvartheta 106, 923, 931, 1020, 1042, 1208, 1217, 1224, 1253, 1279, 1356, 1366, 1373, 1400, 1425	\um@usv@varphi .. 90, 925, 933, 1022, 1044, 1203, 1219, 1226, 1255, 1281, 1351, 1368, 1375, 1402, 1427
\um@usv@Nabla 120, 247, 500, 505, 1013, 1036, 1182, 1186, 1246, 1272, 1331, 1335, 1394, 1419	\um@usv@varpi ... 92, 927, 935, 1024, 1046, 1205, 1221, 1228, 1257, 1283, 1353, 1370, 1377, 1404, 1429
\um@usv@num 23, 582, 1063, 1118, 1121, 1131, 1141, 1151, 1234, 1260, 1301, 1382, 1407	\um@usv@varrho .. 91, 926, 934, 1023, 1045, 1204, 1220, 1227, 1256, 1282, 1352, 1369, 1376, 1403, 1428
\um@usv@partial 126, 265, 502, 506, 1018, 1040, 1199, 1230, 1251, 1277, 1347, 1363, 1398, 1423	\um@usv@varTheta 85, 880, 913, 916, 1014, 1035, 1180, 1188, 1190, 1245, 1271, 1329, 1330, 1337, 1339, 1393, 1418
\um@usv@scrLatin 35, 1066	\um@usv@vartheta 88, 923, 931, 1020, 1042, 1201, 1217, 1224, 1253, 1279, 1349, 1366, 1373, 1400, 1425
\um@usv@scrLatin 36, 1077	\um@zf@feature 673, 685, 688
\um@usv@sfitLatin .. 46, 230, 1098, 1134	\um_bf_Greek_up_or_it_usv . 204, 1185
\um@usv@sfitlatin .. 47, 231, 1106, 1137	\um_bf_greek_up_or_it_usv . 205, 1214
\um@usv@sfitnum 41	\um_bf_Latin_up_or_it_usv . 206, 1158
\um@usv@sflatin 43, 223	\um_bf_latin_up_or_it_usv . 207, 1167
\um@usv@sflatin 45, 224	\um_bfNabla_up_or_it_usv 248, 252, 518, 1186
\um@usv@sfnun 39, 1118, 1121, 1131	\um_bfpartial_up_or_it_usv 266, 270, 520, 1230
\um@usv@sfnun 42, 1097, 1128	\um_bfsf_Greek_up_or_it_usv 227, 234, 1334
\um@usv@sfnun 44, 1105, 1124	\um_bfsf_greek_up_or_it_usv 228, 235, 1362
\um@usv@sfnun 40	\um_bfsf_Latin_up_or_it_usv 225, 232, 1308
\um@usv@ttLatin 49, 1144	
\um@usv@ttlLatin 50, 1147	
\um@usv@ttnum 48, 1141	
\um@usv@upGreek 26, 879, 912, 915, 1012, 1034, 1178, 1185, 1244, 1270, 1327, 1334, 1392, 1417	
\um@usv@upgreek 27, 882, 921, 929, 1017, 1039, 1197, 1214, 1250, 1276, 1345, 1362, 1397, 1422	
\um@usv@upLatin 24, 874, 896, 898, 1005,	

\um_bfsf_latin_up_or_it_usv	\um_config_mathit_greek: 1038
..... 226, 233, 1317	\um_config_mathit_Latin: 1026
\um_bfsfNabla_up_or_it_usv	\um_config_mathit_latin: 1029
..... 249, 253, 519, 1335	\um_config_mathscr_Latin: 1065
\um_bfsfpartial_up_or_it_usv	\um_config_mathscr_latin: 1076
..... 267, 271, 521, 1363	\um_config_mathsf_Latin: 1095
\um_config_mathbb_Latin: 1052	\um_config_mathsf_latin: 1103
\um_config_mathbb_latin: 1048	\um_config_mathsf_num: 1117
\um_config_mathbb_num: 1062	\um_config_mathsf_Latin: 1133
\um_config_mathbf_Greek: 1175	\um_config_mathsf_Latin: 1136
\um_config_mathbf_greek: 1194	\um_config_mathsf_Latin: 1130
\um_config_mathbf_Latin: 1153	\um_config_mathsfup_Latin: 1127
\um_config_mathbf_latin: 1161	\um_config_mathsfup_latin: 1123
\um_config_mathbf_num: 1150	\um_config_mathsfup_num: 1120
\um_config_mathbffrak_Latin: ... 1286	\um_config_mathhtt_Latin: 1143
\um_config_mathbffrak_latin: ... 1289	\um_config_mathhtt_latin: 1146
\um_config_mathbfit_Greek: 1243	\um_config_mathhtt_num: 1140
\um_config_mathbfit_greek: 1248	\um_config_mathup_Greek: 1011
\um_config_mathbfit_Latin: 1236	\um_config_mathup_greek: 1016
\um_config_mathbfit_latin: 1239	\um_config_mathup_Latin: 1004
\um_config_mathbfit_num: 1233	\um_config_mathup_latin: 1007
\um_config_mathbfscr_Latin: ... 1293	\um_glyph_if_exist:n 986
\um_config_mathbfscr_latin: ... 1296	\um_glyph_if_exist:nT 959
\um_config_mathbfssf_Greek: 1325	\um_glyph_if_exist:nTF
\um_config_mathbfssf_greek: 1343 966, 986, 1475, 1478, 1481
\um_config_mathbfssf_Latin: 1303	\um_init_alphabet:n 460, 983
\um_config_mathbfssf_latin: 1311	\um_make_mathactive:nNN 535, <u>537</u>
\um_config_mathbfssf_num: 1300	\um_map_char:nn 606, 609,
\um_config_mathbfssf_Greek: ... 1416	613–618, 621–626, 778, 877, 880,
\um_config_mathbfssf_Greek: ... 1421	904, 907, 913, 916, 922–927, 930–935
\um_config_mathbfssf_Latin: ... 1409	\um_map_char:nn␣ <u>759</u>
\um_config_mathbfssf_latin: ... 1412	\um_map_chars_greek:nn
\um_config_mathbfssf_num: 1406	... 605, 608, 612, 620, 772, 879,
\um_config_mathbfssfup_Greek: ... 1391	881, 882, 889–892, 912, 915, 921, 929
\um_config_mathbfssfup_greek: ... 1396	\um_map_chars_latin:nn
\um_config_mathbfssfup_Latin: ... 1384	595, 597, 600, 602, 769, 874–876,
\um_config_mathbfssfup_latin: ... 1387	878, 885–888, 896, 898, 903, 906
\um_config_mathbfssfup_num: 1381	\um_map_chars_numbers:nn ... 582, 775
\um_config_mathbfup_Greek: 1269	\um_map_chars_range:nnn
\um_config_mathbfup_greek: 1275 759, 770, 773, 776, 779
\um_config_mathbfup_Latin: 1262	\um_mathmap:Nnn 458, 465, 784, 791
\um_config_mathbfup_latin: 1265	\um_mathmap_noparse:Nnn 458, <u>632</u>
\um_config_mathbfup_num: 1259	\um_mathmap_parse:Nnn 465, <u>637</u>
\um_config_mathfrak_Latin: 1083	\um_maybe_init_alphabet:n 460, 467, 960
\um_config_mathfrak_latin: 1091	\um_Nabla_up_or_it_usv .. 247, 251, 505
\um_config_mathit_Greek: 1033	\um_nprimes:n 1465, 1475, 1478, 1481, 1484

\um_nprimes_select:n	1471, 1502	\um_set_mathalphabet_char:Nnnn ..	<u>781</u>
\um_partial_up_or_it_usv	265, 269, 506	\um_set_mathalphabet_greek:Nnn ..	
\um_peek_execute_branches_ss:	803, 1012, 1017, 1034,
.....	1634, 1639		1039, 1178, 1179, 1185, 1197,
\um_peek_execute_branches_ss_aux:			1198, 1214, 1244, 1250, 1270,
.....	1646, 1659		1276, 1327, 1328, 1334, 1345,
\um_prepare_alph:n	961, <u>989</u>		1346, 1362, 1392, 1397, 1417, 1422
\um_process_symbol_noparse:nnnn .		\um_set_mathalphabet_latin:Nnn ..	
.....	457, <u>485</u>	800, 1005,
\um_process_symbol_parse:nnnn	464, <u>485</u>		1008, 1027, 1030, 1049, 1053,
\um_remap_symbol:nnn			1066, 1077, 1084, 1092, 1097,
.....	459, 466, 494, 495, 497,		1098, 1100, 1105, 1106, 1109,
500–503, 505, 506, 509–516, 518–521			1124, 1128, 1134, 1137, 1144,
\um_remap_symbol_noparse:nnn	459, <u>493</u>		1147, 1155, 1156, 1158, 1163,
\um_remap_symbol_parse:nnn			1164, 1167, 1237, 1240, 1249,
.....	466, <u>493</u> , 524		1263, 1266, 1287, 1290, 1294,
\um_remap_symbols:	473, <u>493</u>		1297, 1305, 1306, 1308, 1313,
\um_scan_sscript: 1549, 1582, 1619, 1621			1314, 1317, 1385, 1388, 1410, 1413
\um_scan_sscript:TF	1620, 1629	\um_set_mathalphabet_numbers:Nnn	
\um_scanprime:		797, 1063, 1118, 1121, 1131, 1141,	
.....	1487, 1496, 1507, 1511, 1512		1151, 1234, 1260, 1301, 1382, 1407
\um_scanprime_collect:		\um_set_mathcode:nnnn	<u>411</u> , 531, 648, 762
.....	1489, 1491, 1494, 1497, 1500	\um_setup_active_subscript:nn . . .	
\um_set_delcode:n . . .	547, 550–572, 577	1573, 1587–1614
\um_set_delcode:nn 544–546, 548, 549, 574		\um_setup_active_superscript:nn .	
\um_set_mathalph_range:Nnn	<u>788</u>	1540, 1554–1570
\um_set_mathalph_range:nNnn		\um_setup_alphabets:	477, 938
.....	788, 798, 801, 804	\um_setup_alphanum:	476, <u>580</u>
\um_set_mathalphabet_char:Nnn	782,	\um_setup_bf_literals:	592, <u>884</u>
1009, 1013, 1014, 1018–1024,		\um_setup_bfshapes:	203, 483
1031, 1035, 1036, 1040–1046,		\um_setup_delcodes:	475, <u>543</u> , <u>574</u>
1050, 1054–1060, 1067–1074,		\um_setup_Greek:	588, <u>910</u>
1078–1081, 1085–1089, 1093,		\um_setup_greek:	589, <u>919</u>
1107, 1111, 1113, 1125,		\um_setup_Latin:	586, <u>894</u>
1138, 1148, 1165, 1169,		\um_setup_latin:	587, <u>901</u>
1171, 1176, 1180–1183, 1186,		\um_setup_literals:	584, <u>873</u>
1188, 1190, 1195, 1199–1212,		\um_setup_math_alphabet:nn	
1216–1221, 1223–1228, 1230,		939–955, <u>957</u>
1241, 1245, 1246, 1251–1257,		\um_setup_mathactives:	474, <u>534</u>
1267, 1271–1273, 1277–1284,		\um_setup_mathup:	1673
1291, 1298, 1315, 1319,		\um_setup_nabla:	245, 480
1321, 1329–1332, 1335, 1337,		\um_setup_partial:	263, 481
1339, 1347–1360, 1363,		\um_setup_sfshapes:	221, 482
1365–1370, 1372–1377, 1389,		\um_setup_shapes:	472, 479
1393, 1394, 1398–1404,		\um_sf_Latin_up_or_it_usv	
1414, 1418, 1419, 1423–1429		223, 230, 1100

\um_sf_latin_up_or_it_usv		\version@elt	314
..... 224, 231, 1109		\version@list	313
\um_sub_or_super:n .. 1547, 1580, 1623			
\um_symfont_tl 455, 463, 469,		W	
486, 531, 539, 575, 578, 634, 641, 764		\widehat	1709
\UnicodeMathSymbol 457, 464, 1450		\widetilde	1710
\unless	704		
\updefault	470	X	
\upDigamma	1713	\xdef	667, 700
\updigamma	1712	\XeTeXdelcode	393, 398, 575, 578
\upint	656	\XeTeXdelimiter	392, 397
\Upsilon	828	\XeTeXmathaccent	402
\upsilon	855	\XeTeXmathchardef	376, 538
\use:c	967, 992, 1000	\XeTeXmathcode	394, 399, 404, 412
\use_none:n	467	\XeTeXmathcodenum	541, 1543, 1576
		\XeTeXradical	389
V		\Xi	821
\varepsilon	862	\xi	848
\varkappa	866	\XKV@rm	440
\varointclockwise	653		
\varphi	867	Z	
\varpi	871	\Z	1060, 1089
\varrho	870	\z@	1516, 1519, 1520, 1524
\varsigma	852	\Zeta	813
\varTheta	825	\zeta	840
\vartheta	865	\zf@family	470
\vec	1675	\zf@fontspec	427
		\zf@update@ff	686, 689