Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/09/30 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 `\@onlypreamble` 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

7 Maths alphabets mapping definitions 48
7.1 Non-bold math alphabets 50
7.2 Bold math alphabets 54
7.3 Definitions of the math symbols 60

8 Epilogue 61

II STIX table data extraction 69

A Documenting maths support in the NFSS 71
A.1 Overview 71

III XeTeX math font dimensions 72

1 Introduction

This document describes the unicode-math package, which is an *experimental* implementation of a macro to unicode glyph encoding for mathematical characters. Its intended use is for XeTeX, 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, `\xi` to ξ, `\leq` to $\leq$, etc., `\mathcal{H}` to $\mathcal{H}$ and so on, all for unicode glyphs within a single font.

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

Finally, maths versions must also be provided for. While I guess version selection in LaTeX 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*⟩`,`⟨*font features*⟩`]{`⟨*font name*⟩`}`

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

Not yet implemented: preset names ranges could be used in the range spec., such as `MiscMathSymbolsA`, with such ranges based on unicode chunks. The amount of optimisation required here to achieve acceptable performance has yet to be determined. Techniques such as saving out unicode subsets based on ⟨*unicode range*⟩ data to be `\input` in the next LaTeX run are a possibility, but at this stage, performance without such measures seems acceptable.

2.2 Script and scriptscript fonts/features

Cambria Math uses OpenType font features to activate smaller optical sizes for scriptsize and scriptscriptsize symbols (the B and C, respectively, in A_{B_C}). Other fonts will possibly use entirely separate fonts.

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

3 Maths input

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

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.

	Example	
Package option	Latin	Greek
`math-style=ISO`	(a, z, B, X)	$(\alpha, \beta, \Gamma, \Xi)$
`math-style=TeX`	(a, z, B, X)	$(\alpha, \beta, \Gamma, \Xi)$
`math-style=French`	$(a, z, \mathrm{B}, \mathrm{X})$	$(\alpha, \beta, \Gamma, \Xi)$

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

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

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

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

3.2 Bold style

Similar as in the previous section, ISO standards differ somewhat to TeX's conventions (and classical typesetting) for 'boldness' in mathematics. In the past, it has been customary to use bold *upright* letters to denote things like vectors and matrices. For example, $\mathbf{M} = (M_x, M_y, M_z)$. Presumably, this was due to the relatively

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

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

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

In `unicode-math`, the `\mathbf` 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`, `\mathbfsfup`, and `\mathbfsfit` commands discussed in section §3.4.

How should the generic `\mathsf` behave? Unlike bold, sans serif is used much more sparingly in mathematics. I've seen recommendations to typeset tensors in sans serif italic or sans serif italic bold (e.g., examples in the isomath and mattens packages). But 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 `\mathsfit`, 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	`\mathup`	•	•	•
		Bold	`\mathbfup`	•	•	•
	Italic	Normal	`\mathit`	•	•	•
		Bold	`\mathbfit`	•	•	•
Sans serif	Upright	Normal	`\mathsfup`	•		•
	Italic	Normal	`\mathsfit`	•		•
	Upright	Bold	`\mathsfbfup`	•	•	•
	Italic	Bold	`\mathsfbfit`	•	•	•
Typewriter	Upright	Normal	`\mathtt`	•		•
Double-struck	Upright	Normal	`\mathbb`	•		•
Script	Upright	Normal	`\mathscr`	•		
		Bold	`\matbfscr`	•		
Fraktur	Upright	Normal	`\mathfrak`	•		
		Bold	`\mathbffrac`	•		

Table 4: The various forms of nabla.

Description		Glyph
Upright	Serif	∇
	Bold serif	$\boldsymbol{\nabla}$
	Bold sans	∇
Italic	Serif	$\mathit{\nabla}$
	Bold serif	$\boldsymbol{\mathit{\nabla}}$
	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	$\boldsymbol{\partial}$
	Italic	$\boldsymbol{\partial}$
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`.

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

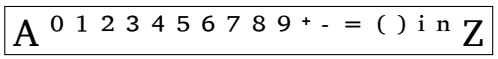


Figure 1: The unicode superscripts supported as input characters. These are the literal glyphs from Charis SIL, not the output seen when used for maths input. The 'A' and 'Z' are to provide context for the size and location of the superscript glyphs.

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`, `\primetriple`, and `\primequadruple`.

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

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

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: '$a\colon b$'.

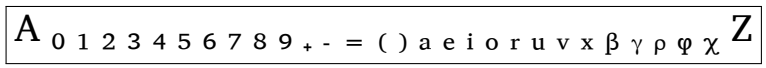


Figure 2: The unicode subscripts supported as input characters. See note from figure 1.

Table 6: Slashes and backslashes.

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

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 $\sum$).

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 LaTeX 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': $\pi \approx 7 \operatorname{⧵} 22$.[3] The LaTeX name for this character is `\setminus`.

Finally, U+29F9: BIG REVERSE SOLIDUS is a 'big operator' (like $\sum$).

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

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

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 `\left`fracslash 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 ▯▯▯▯://▯▯▯.▯3.▯▯▯/▯▯/▯▯▯▯▯▯3/

[3] This 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.

```
\ProvidesPackage{unicode-math}
  [2009/09/30 v0.4 Unicode maths in XeLaTeX]
```

4 Things we need

Packages

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

Start using LaTeX3 — finally!

```
\ExplSyntaxOn
```

Counters and conditionals

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

For `math-style`:

```
\newif\if@um@literal
\newif\if@um@upGreek
\newif\if@um@upgreek
\newif\if@um@upLatin
\newif\if@um@uplatin
```

For `bold-style`:

```
\newif\if@um@bfliteral
\newif\if@um@bfupGreek
\newif\if@um@bfupgreek
\newif\if@um@bfupLatin
\newif\if@um@bfuplatin
```

For `nabla`:

```
\newif\if@um@upNabla
\newif\if@um@uppartial
\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]

```
\def\um@usv@num{`\0}
\def\um@usv@upLatin{`\A}
\def\um@usv@uplatin{`\a}
\def\um@usv@upGreek{"391}
\def\um@usv@upgreek{"3B1}
\def\um@usv@itLatin{"1D434}
\def\um@usv@itlatin{"1D44E}
\def\um@usv@itGreek{"1D6E2}
\def\um@usv@itgreek{"1D6FC}
\def\um@usv@bbnum{"1D7D8}
\def\um@usv@bbLatin{"1D538}
\def\um@usv@bblatin{"1D552}
\def\um@usv@scrLatin{"1D49C}
\def\um@usv@scrlatin{"1D4B6}
\def\um@usv@frakLatin{"1D504}
\def\um@usv@fraklatin{"1D51E}
\def\um@usv@sfnum{"1D7E2}
\def\um@usv@sfupnum{"1D7E2}
```

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

```
\def\um@usv@sfitnum{"1D7E2}
\def\um@usv@sfupLatin{"1D5A0}
\def\um@usv@sfLatin  {"1D5A0}
\def\um@usv@sfuplatin{"1D5BA}
\def\um@usv@sflatin{"1D5BA}
\def\um@usv@sfitLatin{"1D608}
\def\um@usv@sfitlatin{"1D622}
\def\um@usv@ttnum{"1D7F6}
\def\um@usv@ttLatin{"1D670}
\def\um@usv@ttlatin{"1D68A}
```

Bold:

```
\def\um@usv@bfnum  {"1D7CE}
\def\um@usv@bfupnum{"1D7CE}
\def\um@usv@bfitnum{"1D7CE}
\def\um@usv@bfupLatin{"1D400}
\def\um@usv@bfLatin  {"1D400}
\def\um@usv@bfuplatin{"1D41A}
\def\um@usv@bflatin  {"1D41A}
\def\um@usv@bfupGreek{"1D6A8}
\def\um@usv@bfupgreek{"1D6C2}
\def\um@usv@bfGreek  {"1D6A8}
\def\um@usv@bfgreek  {"1D6C2}
\def\um@usv@bfitLatin{"1D468}
\def\um@usv@bfitlatin{"1D482}
\def\um@usv@bfitGreek{"1D71C}
\def\um@usv@bfitgreek{"1D736}
\def\um@usv@bffrakLatin{"1D56C}
\def\um@usv@bffraklatin{"1D586}
\def\um@usv@bfscrLatin{"1D4D0}
\def\um@usv@bfscrlatin{"1D4EA}
\def\um@usv@bfsfnum  {"1D7EC}
\def\um@usv@bfsfupnum{"1D7EC}
\def\um@usv@bfsfitnum{"1D7EC}
\def\um@usv@bfsfupLatin{"1D5D4}
\def\um@usv@bfsfLatin  {"1D5D4}
\def\um@usv@bfsfuplatin{"1D5EE}
\def\um@usv@bfsflatin  {"1D5EE}
\def\um@usv@bfsfupGreek{"1D756}
\def\um@usv@bfsfupgreek{"1D770}
\def\um@usv@bfsfGreek  {"1D756}
\def\um@usv@bfsfgreek  {"1D770}
\def\um@usv@bfsfitLatin{"1D63C}
\def\um@usv@bfsfitlatin{"1D656}
\def\um@usv@bfsfitGreek{"1D790}
\def\um@usv@bfsfitgreek{"1D7AA}
```

Greek variants:

```
\def\um@usv@varTheta{"3F4}
\def\um@usv@Digamma{"3DC}
\def\um@usv@varepsilon{"3F5}
\def\um@usv@vartheta{"3D1}
\def\um@usv@varkappa{"3F0}
\def\um@usv@varphi{"3D5}
\def\um@usv@varrho{"3F1}
\def\um@usv@varpi{"3D6}
\def\um@usv@digamma{"3DD}
```

Bold:

```
\def\um@usv@bfvarTheta{"1D6B9}
\def\um@usv@bfDigamma{"1D7CA}
\def\um@usv@bfvarepsilon{"1D6DC}
\def\um@usv@bfvartheta{"1D6DD}
\def\um@usv@bfvarkappa{"1D6DE}
\def\um@usv@bfvarphi{"1D6DF}
\def\um@usv@bfvarrho{"1D6E0}
\def\um@usv@bfvarpi{"1D6E1}
\def\um@usv@bfdigamma{"1D7CB}
```

Italic Greek variants:

```
\def\um@usv@ith{"210E}
\def\um@usv@itvarTheta{"1D6F3}
\def\um@usv@itvarepsilon{"1D716}
\def\um@usv@itvartheta{"1D717}
\def\um@usv@itvarkappa{"1D718}
\def\um@usv@itvarphi{"1D719}
\def\um@usv@itvarrho{"1D71A}
\def\um@usv@itvarpi{"1D71B}
```

Bold italic:

```
\def\um@usv@bfuph{"1D421}
\def\um@usv@bfith{"1D489}
\def\um@usv@bfitvarTheta{"1D72D}
\def\um@usv@bfitvarepsilon{"1D750}
\def\um@usv@bfitvartheta{"1D751}
\def\um@usv@bfitvarkappa{"1D752}
\def\um@usv@bfitvarphi{"1D753}
\def\um@usv@bfitvarrho{"1D754}
\def\um@usv@bfitvarpi{"1D755}
```

Nabla:

```
\def\um@usv@Nabla{"2207}
\def\um@usv@itNabla{"1D6FB}
\def\um@usv@bfNabla{"1D6C1}
\def\um@usv@bfitNabla{"1D735}
\def\um@usv@bfsfNabla{"1D76F}
\def\um@usv@bfsfitNabla{"1D7A9}
```

Partial:

```
\def\um@usv@partial{"2202}
\def\um@usv@itpartial{"1D715}
\def\um@usv@bfpartial{"1D6DB}
\def\um@usv@bfitpartial{"1D74F}
\def\um@usv@bfsfpartial{"1D789}
\def\um@usv@bfsfitpartial{"1D7C3}
```

4.1 Package options

xkeyval's package support is used here.

math-style

```
\define@choicekey*{unicode-math.sty}
    {math-style}[\@tempa\@tempb]{iso,tex,french,literal}{
  \ifcase\@tempb\relax
    \@um@upGreekfalse
    \@um@upgreekfalse
    \@um@upLatinfalse
    \@um@uplatinfalse
    \@um@bfupGreekfalse
    \@um@bfupgreekfalse
    \@um@uppartialfalse
    \@um@bfupLatinfalse
    \@um@bfuplatinfalse
    \@um@upNablafalse
    \bool_set_false:N \g_um_upsans_bool
    \bool_set_false:N \g_um_texgreek_bool
  \or
    \@um@upGreektrue
    \@um@upgreekfalse
    \@um@upLatinfalse
    \@um@uplatinfalse
    \@um@bfupGreektrue
    \@um@bfupgreekfalse
    \@um@uppartialfalse
    \@um@bfupLatintrue
    \@um@bfuplatintrue
    \@um@upNablatrue
    \bool_set_true:N \g_um_upsans_bool
    \bool_set_true:N \g_um_texgreek_bool
  \or
    \@um@upGreektrue
    \@um@upgreektrue
    \@um@upLatintrue
```

```
    \@um@uplatinfalse
    \@um@bfupGreektrue
    \@um@bfupgreektrue
    \@um@uppartialtrue
    \@um@bfupLatintrue
    \@um@bfuplatintrue
    \@um@upNablatrue
    \bool_set_true:N \g_um_upsans_bool
    \bool_set_false:N \g_um_texgreek_bool
  \or
    \@um@literaltrue
    \@um@bfliteraltrue
    \bool_set_true:N \g_um_sfliteral_bool
    \bool_set_false:N \g_um_texgreek_bool
  \fi
}
```

bold-style

```
\define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,french,literal}{
  \ifcase\@tempb\relax
    \@um@bfupGreekfalse
    \@um@bfupgreekfalse
    \@um@bfupLatinfalse
    \@um@bfuplatinfalse
    \@um@uppartialfalse
  \or
    \@um@bfupGreektrue
    \@um@bfupgreekfalse
    \@um@bfupLatintrue
    \@um@bfuplatintrue
    \@um@uppartialfalse
  \or
    \@um@bfupGreektrue
    \@um@bfupgreektrue
    \@um@bfupLatintrue
    \@um@bfuplatintrue
    \@um@uppartialtrue
  \or
    \@um@bfliteraltrue
  \fi
}
\cs_set:Nn \um_setup_bfshapes: {
  \tl_set:Nx \um_bf_Greek_up_or_it_usv { \if@um@bfupGreek \um@usv@bfupGreek \else \um@usv@bfit
  \tl_set:Nx \um_bf_greek_up_or_it_usv { \if@um@bfupgreek \um@usv@bfupgreek \else \um@usv@bfit
  \tl_set:Nx \um_bf_Latin_up_or_it_usv { \if@um@bfupLatin \um@usv@bfupLatin \else \um@usv@bfit
  \tl_set:Nx \um_bf_latin_up_or_it_usv { \if@um@bfuplatin \um@usv@bfuplatin \else \um@usv@bfit
```

```
}
```

sans-style

```
\bool_new:N \g_um_upsans_bool
\bool_new:N \g_um_sfliteral_bool
\define@choicekey*{unicode-math.sty}
    {sans-style}[\@tempa\@tempb]{iso,tex,literal}{
  \ifcase\@tempb\relax
    \bool_set_false:N \g_um_upsans_bool
  \or
    \bool_set_true:N \g_um_upsans_bool
  \or
    \bool_set_true:N \g_um_sfliteral_bool
  \fi
}
\cs_set:Nn \um_setup_sfshapes: {
  \bool_if:NTF \g_um_upsans_bool {
    \tl_set:Nn \um_sf_Latin_up_or_it_usv       { \um@usv@sfLatin     }
    \tl_set:Nn \um_sf_latin_up_or_it_usv       { \um@usv@sflatin     }
    \tl_set:Nn \um_bfsf_Latin_up_or_it_usv     { \um@usv@bfsfupLatin }
    \tl_set:Nn \um_bfsf_latin_up_or_it_usv     { \um@usv@bfsfuplatin }
    \tl_set:Nn \um_bfsf_Greek_up_or_it_usv     { \um@usv@bfsfupGreek }
    \tl_set:Nn \um_bfsf_greek_up_or_it_usv     { \um@usv@bfsfupgreek }
  }{
    \tl_set:Nn \um_sf_Latin_up_or_it_usv       { \um@usv@sfitLatin   }
    \tl_set:Nn \um_sf_latin_up_or_it_usv       { \um@usv@sfitlatin   }
    \tl_set:Nn \um_bfsf_Latin_up_or_it_usv     { \um@usv@bfsfitLatin }
    \tl_set:Nn \um_bfsf_latin_up_or_it_usv     { \um@usv@bfsfitlatin }
    \tl_set:Nn \um_bfsf_Greek_up_or_it_usv     { \um@usv@bfsfitGreek }
    \tl_set:Nn \um_bfsf_greek_up_or_it_usv     { \um@usv@bfsfitgreek }
  }
}
```

Symbol obliqueness

```
\define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
  \ifcase\@tempb\relax
    \@um@upNablatrue
  \or
    \@um@upNablafalse
  \fi
}
\cs_set:Nn \um_setup_nabla: {
  \if@um@upNabla
    \tl_set:Nn \um_Nabla_up_or_it_usv     { \um@usv@Nabla }
    \tl_set:Nn \um_bfNabla_up_or_it_usv   { \um@usv@bfNabla }
```

```
    \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfNabla }
  \else
    \tl_set:Nn \um_Nabla_up_or_it_usv     { \um@usv@itNabla }
    \tl_set:Nn \um_bfNabla_up_or_it_usv   { \um@usv@bfitNabla }
    \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfitNabla }
  \fi
}
\define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
  \ifcase\@tempb\relax
    \@um@uppartialtrue
  \or
    \@um@uppartialfalse
  \fi
}
\cs_set:Nn \um_setup_partial: {
  \if@um@uppartial
    \tl_set:Nn \um_partial_up_or_it_usv     { \um@usv@partial }
    \tl_set:Nn \um_bfpartial_up_or_it_usv   { \um@usv@bfpartial }
    \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfpartial }
  \else
    \tl_set:Nn \um_partial_up_or_it_usv     { \um@usv@itpartial }
    \tl_set:Nn \um_bfpartial_up_or_it_usv   { \um@usv@bfitpartial }
    \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfitpartial }
  \fi
}
```

Epsilon and phi shapes

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

Colon style

```
\bool_new:N \g_um_literal_colon_bool
\define@choicekey*{unicode-math.sty}{colon}[\@tempa\@tempb]{literal,TeX}{
  \ifcase\@tempb\relax
    \bool_set_true:N \g_um_literal_colon_bool
  \or
    \bool_set_false:N \g_um_literal_colon_bool
  \fi
}
```

Slash delimiter style

```
\define@choicekey*{unicode-math.sty}{slash-delimiter}[\@tempa\@tempb]{ascii,frac,div}{
  \ifcase\@tempb\relax
    \tl_set:Nn \g_um_slash_delimiter_usv {"002F}
  \or
    \tl_set:Nn \g_um_slash_delimiter_usv {"2044}
  \or
    \tl_set:Nn \g_um_slash_delimiter_usv {"2215}
  \fi
}
\ExecuteOptionsX{math-style=TeX,slash-delimiter=ascii}
\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.

```
\tl_map_inline:nn {
\new@mathgroup
\cdp@list
\cdp@elt
\DeclareMathSizes
\@DeclareMathSizes
\newmathalphabet
\newmathalphabet@@
\newmathalphabet@@@
\DeclareMathVersion
\define@mathalphabet
\define@mathgroup
\addtoversion
\version@list
\version@elt
\alpha@list
\alpha@elt
\restore@mathversion
\init@restore@version
\dorestore@version
\process@table
\new@mathversion
\DeclareSymbolFont
\group@list
\group@elt
\new@symbolfont
\SetSymbolFont
\SetSymbolFont@
```

```
\get@cdp
\DeclareMathAlphabet
\new@mathalphabet
\SetMathAlphabet
\SetMathAlphabet@
\DeclareMathAccent
\set@mathaccent
\DeclareMathSymbol
\set@mathchar
\set@mathsymbol
\DeclareMathDelimiter
\@xxDeclareMathDelimiter
\@DeclareMathDelimiter
\@xDeclareMathDelimiter
\set@mathdelimiter
\set@@mathdelimiter
\DeclareMathRadical
\mathchar@type
\DeclareSymbolFontAlphabet
\DeclareSymbolFontAlphabet@
}{
  \tl_remove_in:Nn \@preamblecmds {\do#1}
}
```

4.3 Other things

`\um@fontdimen@percent` `#1` : Font dimen number

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

0.73
0.60
0.65

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

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

`\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.

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

```
  \ifx#1\scriptstyle
    #2\um@fontdimen@percent{10}\um@font#3
  \else
    \ifx#1\scriptscriptstyle
      #2\um@fontdimen@percent{11}\um@font#3
    \else
      #2#3%
    \fi
  \fi
}
```

5 Fundamentals

5.1 Enlarging the number of maths families

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

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

This is sufficient for LaTeX'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`

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

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

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

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

```
\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`.

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

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

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

```
  \cs_gset:cpn { \cs_to_str:N #2 op } {
    \csname\string#2@sym\endcsname
    \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
    \ifin@
      \expandafter\nolimits
    \fi
  }
```

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

```
\else
```

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

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

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

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

```
        \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
      \else
        \ifx\mathclose#3\relax
          \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
          \global\XeTeXdelcode#4=#1 #4\relax
          \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
        \else
```

Accents

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

```
            \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
          \fi
        \fi
      \fi
    \fi
  \fi
}
```

`\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.

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

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

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

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

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

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

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

- Tell fontspec that maths font features are actually allowed.

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

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

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

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

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

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

```
\tl_set:Nn \l_um_script_features_tl  {ScriptStyle}
\tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
\tl_set:Nn \l_um_script_font_tl      {#2}
\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.

```
\setkeys*[um]{options}{#1}
\edef\@tempa{\noexpand\zf@fontspec{
    Script = Math,
    SizeFeatures = {
      {Size = \tf@size-} ,
      {Size = \sf@size-\tf@size ,
       Font = \l_um_script_font_tl ,
       \l_um_script_features_tl
      } ,
      {Size = -\sf@size ,
       Font = \l_um_sscript_font_tl ,
       \l_um_sscript_features_tl
      }
    },
    \XKV@rm
  }{#2}
}
\@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`:

```
  \font\um@font="#2"\relax
%%  \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
%%    \@um@ot@math@true
%%  \else
%%    \PackageWarningNoLine{unicode-math}{
%%      The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
%%      Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
%%      in~ a~ substandard~ manner
%%    }
%%  \fi
```

If we're defining the full unicode math repetoire, 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

```
  \ifx\um@char@range\@empty
    \tl_set:Nn \um_symfont_tl {um@allsym}
   \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
    \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
    \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
    \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
    \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
  \else
    \stepcounter{um@fam}
    \tl_set:Nx \um_symfont_tl {um@fam\theum@fam}
    \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn
    \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
    \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
    \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
  \fi
```

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

```
  \DeclareSymbolFont{\um_symfont_tl}
    {\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`.

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

Finally,

- Set up shapes for italic/upright or ordinary/var symbols as per package options.

- Remap symbols that don't take their natural mathcode
- Activate any symbols that need to be math-active
- Setup all symbols not covered by the table (mostly alphanumerics)
- Setup the maths alphabets (`\mathbf` etc.)

```
  \um_setup_shapes:
  \um_remap_symbols:
  \um_setup_mathactives:
  \um_setup_delcodes:
  \um_setup_alphanum:
  \um_setup_alphabets:
```

End of the `\setmathfont` macro.

```
}

\cs_new:Nn \um_setup_shapes: {
  \um_setup_nabla:
  \um_setup_partial:
  \um_setup_sfshapes:
  \um_setup_bfshapes:
}
```

5.3.1 Functions for setting up symbols with mathcodes

`\um_process_symbol_noparse:nnnn`
`\um_process_symbol_parse: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.

```
\cs_set:Nn \um_process_symbol_noparse:nnnn {
  \um@mathsymbol{#2}{#3}{\um_symfont_tl}{#1}
}

\cs_set:Nn \um_process_symbol_parse:nnnn {
  \um@parse@term{#1}{#2}{#3}{
    \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
  }
}
```

`\um_remap_symbols:`
`\um_remap_symbol_noparse:nnn`
`\um_remap_symbol_parse:nnn`

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

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

```
  \if@um@literal
    \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
    \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
    \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
    \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
  \else
   \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um_Nabla_up_or_it_usv}
   \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um_partial_up_or_it_u
  \fi
```

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

```
  \if@um@bfliteral
   \um_remap_symbol:nnn {\um@usv@bfNabla     }{\mathord}{\um@usv@bfNabla}
   \um_remap_symbol:nnn {\um@usv@bfitNabla   }{\mathord}{\um@usv@bfitNabla}
   \um_remap_symbol:nnn {\um@usv@bfsfNabla   }{\mathord}{\um@usv@bfsfNabla}
   \um_remap_symbol:nnn {\um@usv@bfsfitNabla }{\mathord}{\um@usv@bfsfitNabla}
   \um_remap_symbol:nnn {\um@usv@bfpartial   }{\mathord}{\um@usv@bfpartial}
   \um_remap_symbol:nnn {\um@usv@bfitpartial }{\mathord}{\um@usv@bfitpartial}
   \um_remap_symbol:nnn {\um@usv@bfsfpartial }{\mathord}{\um@usv@bfsfpartial}
   \um_remap_symbol:nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
  \else
   \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla}{\mathord}{\um_bfNabla_up_or_it_u
   \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla}{\mathord}{\um_bfsfNabla_up_o
   \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um_bfpartial_up_o
   \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um_bfsfpartia
  \fi
}
```

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

```
\cs_new:Nn \um_remap_symbol_parse:nnn {
  \um@parse@term {#3} {\@nil} {#2} {
    \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
  }
}
\cs_new:Nn \um_remap_symbol_noparse:nnn {
  \clist_map_inline:nn {#1} {
    \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
  }
}
```

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:`

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

`\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!

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

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:`

```
\cs_new:Nn \um_setup_delcodes: {
  \um_set_delcode:nn {`\/}   {\g_um_slash_delimiter_usv}
  \um_set_delcode:nn {"2044} {\g_um_slash_delimiter_usv} % fracslash
  \um_set_delcode:nn {"2215} {\g_um_slash_delimiter_usv} % divslash
  \um_set_delcode:n {"005C} % backslash
  \um_set_delcode:nn {`\<} {"27E8} % angle brackets with ascii notation
  \um_set_delcode:nn {`\>} {"27E9} % angle brackets with ascii notation
  \um_set_delcode:n {"2191} % up arrow
  \um_set_delcode:n {"2193} % down arrow
  \um_set_delcode:n {"2195} % updown arrow
  \um_set_delcode:n {"219F} % up arrow twohead
  \um_set_delcode:n {"21A1} % down arrow twohead
  \um_set_delcode:n {"21A5} % up arrow from bar
  \um_set_delcode:n {"21A7} % down arrow from bar
  \um_set_delcode:n {"21A8} % updown arrow from bar
  \um_set_delcode:n {"21BE} % up harpoon right
  \um_set_delcode:n {"21BF} % up harpoon left
  \um_set_delcode:n {"21C2} % down harpoon right
  \um_set_delcode:n {"21C3} % down harpoon left
  \um_set_delcode:n {"21C5} % arrows up down
  \um_set_delcode:n {"21F5} % arrows down up
  \um_set_delcode:n {"21C8} % arrows up up
  \um_set_delcode:n {"21CA} % arrows down down
  \um_set_delcode:n {"21D1} % double up arrow
  \um_set_delcode:n {"21D3} % double down arrow
```

```
  \um_set_delcode:n {"21D5} % double updown arrow
  \um_set_delcode:n {"21DE} % up arrow double stroke
  \um_set_delcode:n {"21DF} % down arrow double stroke
  \um_set_delcode:n {"21E1} % up arrow dashed
  \um_set_delcode:n {"21E3} % down arrow dashed
}
```

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

```
\cs_new:Nn \um_set_delcode:nn {
  \XeTeXdelcode#1 = \csname sym\um_symfont_tl\endcsname #2
}
\cs_new:Nn \um_set_delcode:n {
  \XeTeXdelcode#1 = \csname sym\um_symfont_tl\endcsname #1
}
```

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`.

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

Normal weight

```
    \if@um@literal
      \um_setup_literals:
    \else
      \um_setup_Latin:
      \um_setup_latin:
      \um_setup_Greek:
      \um_setup_greek:
    \fi
```

Bold

```
    \if@um@bfliteral
      \um_setup_bf_literals:
    \else
      \if@um@bfupLatin
     \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfupLatin}
      \else
     \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
      \fi
```

```
      \if@um@bfuplatin
      \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfuplatin}
      \else
      \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
      \fi
      \if@um@bfupGreek
      \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfupGreek}
      \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
      \else
      \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}
      \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
      \fi
      \if@um@bfupgreek
      \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
      \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
      \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
      \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
      \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
      \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
      \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
      \else
      \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
      \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
      \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
      \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
      \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
      \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
      \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
      \fi
    \fi
  \else
```

: TODO : what is supposed to happen here?

```
  \fi
}
```

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 '$\mathbb{A}$'

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.

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

```
      \exp_args:No \um@addto@mathmap \um_symfont_tl {##1}{#1}{#3}
    }
  }
```

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

```
\cs_set:Nn \um_mathmap_parse:Nnn {
  \clist_map_inline:Nn \um@char@num@range {
    \ifnum##1=#3\relax
      \clist_map_inline:nn {#2} {
        \exp_args:No \um@addto@mathmap \um_symfont_tl {####1}{#1}{#3}
      }
    \fi
  }
}
```

`\um@addto@mathmap`

`#1` : Math symbol font, always/usually the expansion of `\um_symfont_tl`
`#2` : Input slot, *e.g.*, the slot for 'A'
`#3` : Maths alphabet, *e.g.*, `\mathbb`
`#4` : Output slot, *e.g.*, the slot for '𝔸'

This macro is used so that `\um_symfont_tl` can be expanded before entering the `\g@addto@macro` command.

```
\newcommand\um@addto@mathmap[4]{
  \tl_put_right:cn {um_setup_\cs_to_str:N #3:} {
    \um_set_mathcode:nnnn{#2}{\mathalpha}{#1}{#4}
  }
}
```

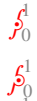
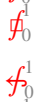
5.4 (Big) operators

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.*, `\def\int{\intop\nolimits}`, so there needs to be a transformation from `\int` to `\intop` during the expansion of `\UnicodeMathSymbol` in the appropriate contexts.

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

USV	Ex.	Macro	Description
U+02140		\Bbbsum	DOUBLE-STRUCK N-ARY SUMMATION
U+0220F		\prod	PRODUCT OPERATOR
U+02210		\coprod	COPRODUCT OPERATOR
U+02211		\sum	SUMMATION OPERATOR
U+0222B		\int	INTEGRAL OPERATOR
U+0222C		\iint	DOUBLE INTEGRAL OPERATOR
U+0222D		\iiint	TRIPLE INTEGRAL OPERATOR
U+0222E		\oint	CONTOUR INTEGRAL OPERATOR
U+0222F		\oiint	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230		\oiiint	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231		\intclockwise	CLOCKWISE INTEGRAL
U+02232		\varointclockwise	CONTOUR INTEGRAL, CLOCKWISE
U+02233		\ointctrclockwise	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0		\bigwedge	LOGICAL OR OPERATOR
U+022C1		\bigvee	LOGICAL AND OPERATOR
U+022C2		\bigcap	INTERSECTION OPERATOR
U+022C3		\bigcup	UNION OPERATOR
U+027D5		\leftouterjoin	LEFT OUTER JOIN
U+027D6		\rightouterjoin	RIGHT OUTER JOIN
U+027D7		\fullouterjoin	FULL OUTER JOIN
U+027D8		\bigbot	LARGE UP TACK
U+027D9		\bigtop	LARGE DOWN TACK
U+029F8		\xsol	BIG SOLIDUS
U+029F9		\xbsol	BIG REVERSE SOLIDUS
U+02A00		\bigodot	N-ARY CIRCLED DOT OPERATOR

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

U+02A1E	◁	\bigtriangleleft	LARGE LEFT TRIANGLE OPERATOR
U+02A1F	⨟	\zcmp	Z NOTATION SCHEMA COMPOSITION
U+02A20	≫	\zpipe	Z NOTATION SCHEMA PIPING
U+02A21	↾	\zproject	Z NOTATION SCHEMA PROJECTION
U+02AFC	⫼	\biginterleave	LARGE TRIPLE VERTICAL BAR OPERATOR
U+02AFF	⨿	\bigtalloblong	N-ARY WHITE VERTICAL BAR

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

```
\def\um@nolimits{
 \@elt\int\@elt\iint\@elt\iiint\@elt\iiiint\@elt\oint\@elt\oiint\@elt\oiiint
 \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
 \@elt\intbar\@elt\intBar\@elt\fint\@elt\cirfnint\@elt\awint\@elt\rppolint
 \@elt\scpolint\@elt\npolint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
  \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
}
```

\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.

```
\newcommand\addnolimits[1]{
 \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
}
```

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

```
\def\removenolimits#1{
  \begingroup
    \def\@elt##1{
      \ifx##1#1\else
        \noexpand\@elt\noexpand##1
     \fi}
    \xdef\um@nolimits{\um@nolimits}
  \endgroup
}
```

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.

```
\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`

```
\let\left@primitive\left
\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	(	\lparen	LEFT PARENTHESIS
U+0005B	[	\lbrack	LEFT SQUARE BRACKET
U+0007B	{	\lbrace	LEFT CURLY BRACKET
U+0007C	|	\lvert	VERTICAL BAR
U+02016	‖	\lVert	DOUBLE VERTICAL BAR
U+0221A	√	\sqrt	RADICAL
U+0221B	∛	\cuberoot	CUBE ROOT
U+0221C	∜	\fourthroot	FOURTH ROOT
U+02308	⌈	\lceil	LEFT CEILING
U+0230A	⌊	\lfloor	LEFT FLOOR
U+0231C	⌜	\ulcorner	UPPER LEFT CORNER
U+0231E	⌞	\llcorner	LOWER LEFT CORNER
U+02772		\lbrbrak	LIGHT LEFT TORTOISE SHELL BRACKET ORNAMENT

U+027C5	⟅	\lbag	LEFT S-SHAPED BAG DELIMITER
U+027CC	⟌	\longdivision	LONG DIVISION MATHEMATICAL LEFT WHITE SQUARE
U+027E6	⟦	\lBrack	BRACKET
U+027E8	⟨	\langle	MATHEMATICAL LEFT ANGLE BRACKET MATHEMATICAL LEFT DOUBLE ANGLE
U+027EA	⟪	\lAngle	BRACKET MATHEMATICAL LEFT WHITE TORTOISE
U+027EC		\Lbrbrak	SHELL BRACKET
U+02983	⦃	\lBrace	LEFT WHITE CURLY BRACKET
U+02985	⦅	\lParen	LEFT WHITE PARENTHESIS
U+02987	⦇	\llparenthesis	Z NOTATION LEFT IMAGE BRACKET
U+02989	⦉	\llangle	Z NOTATION LEFT BINDING BRACKET
U+0298B	⦋	\lbrackubar	LEFT SQUARE BRACKET WITH UNDERBAR LEFT SQUARE BRACKET WITH TICK IN TOP
U+0298D	⦍	\lbrackultick	CORNER LEFT SQUARE BRACKET WITH TICK IN
U+0298F	⦏	\lbrackltick	BOTTOM CORNER
U+02991	⦑	\langledot	LEFT ANGLE BRACKET WITH DOT
U+02993	⦓	\lparenless	LEFT ARC LESS-THAN BRACKET
U+02997	⦗	\lblkbrbrak	LEFT BLACK TORTOISE SHELL BRACKET
U+029D8	⧘	\lvzigzag	LEFT WIGGLY FENCE
U+029DA	⧚	\Lvzigzag	LEFT DOUBLE WIGGLY FENCE
U+029FC	⧼	\lcurvyangle	LEFT POINTING CURVED ANGLE BRACKET
U+03014		\lbrbrak	LEFT BROKEN BRACKET
U+03018		\Lbrbrak	LEFT WHITE TORTOISE SHELL BRACKET

And \mathclose:

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

			MATHEMATICAL RIGHT DOUBLE ANGLE
U+027EB	⟫	\rAngle	BRACKET MATHEMATICAL RIGHT WHITE TORTOISE
U+027ED		\Rbrbrak	SHELL BRACKET
U+02984	⦄	\rBrace	RIGHT WHITE CURLY BRACKET
U+02986	⦆	\rParen	RIGHT WHITE PARENTHESIS
U+02988	⦈	\rrparenthesis	Z NOTATION RIGHT IMAGE BRACKET
U+0298A	⦊	\rrangle	Z NOTATION RIGHT BINDING BRACKET
U+0298C	⦌	\rbrackubar	RIGHT SQUARE BRACKET WITH UNDERBAR RIGHT SQUARE BRACKET WITH TICK IN
U+0298E	⦎	\rbracklrtick	BOTTOM CORNER RIGHT SQUARE BRACKET WITH TICK IN TOP
U+02990	⦐	\rbrackurtick	CORNER
U+02992	⦒	\rangledot	RIGHT ANGLE BRACKET WITH DOT
U+02994	⦔	\rparengtr	RIGHT ARC GREATER-THAN BRACKET
U+02998	⦘	\rblkbrbrak	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	$\grave{x}$	\grave	GRAVE ACCENT
U+00301	$\acute{x}$	\acute	ACUTE ACCENT
U+00302	$\hat{x}$	\hat	CIRCUMFLEX ACCENT
U+00303	$\tilde{x}$	\tilde	TILDE
U+00304	$\bar{x}$	\bar	MACRON
U+00305	$\overline{x}$	\overbar	OVERBAR EMBELLISHMENT
U+00306	$\breve{x}$	\breve	BREVE
U+00307	$\dot{x}$	\dot	DOT ABOVE
U+00308	$\ddot{x}$	\ddot	DIERESIS
U+00309	$\dot{x}$	\ovhook	COMBINING HOOK ABOVE
U+0030A	$\mathring{x}$	\ocirc	RING
U+0030C	$\check{x}$	\check	CARON
U+00310	$\candra{x}$	\candra	CANDRABINDU (NON-SPACING)
U+00312	$\oturnedcomma{x}$	\oturnedcomma	COMBINING TURNED COMMA ABOVE
U+00313	$\osmooth{x}$	\osmooth	GREEK PSILI (SMOOTH BREATHING) (NON-SPACING)

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

```
\newcommand\um@zf@feature[2]{
  \define@key[zf]{options}{#1}[]{
    \if@um@fontspec@feature
      #2
    \else
      \PackageError{fontspec/unicode-math}
        {The ‘#1’ font feature can only be used for maths fonts}
        {The feature you tried to use can only be in commands
          like \protect\setmathfont}
    \fi
  }
}
```

6.1 OpenType maths font features

```
\um@zf@feature{ScriptStyle}{
  \zf@update@ff{+ssty=0}
}
\um@zf@feature{ScriptScriptStyle}{
  \zf@update@ff{+ssty=1}
}
```

6.2 Script and scriptscript font options

```
\define@cmdkey[um]{options}[um@]{ScriptFeatures}{}
\define@cmdkey[um]{options}[um@]{ScriptScriptFeatures}{}
\define@cmdkey[um]{options}[um@]{ScriptFont}{}
\define@cmdkey[um]{options}[um@]{ScriptScriptFont}{}
```

6.3 Range processing

The 'ALL' branch here is deprecated and happens automatically.

```
\define@choicekey+[um]{options}{Range}[\@tempa\@tempb]{ALL}{
  \ifcase\@tempb\relax
    \global\let\um@char@range\@empty
  \fi
}{
  \xdef\um@char@range{#1}
}
```

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:

```
\newcommand\um@parse@term[4]{
  \clist_map_variable:NNn \um@char@range \@ii {
    \unless\ifx\@ii\@empty
      \@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`):

```
      \expandafter\um@firstchar\expandafter{\@ii}
      \ifx\@tempa\um@backslash
        \expandafter\ifx\@ii#2\relax
          \@tempswatrue
        \else
          \expandafter\ifx\@ii#3\relax
            \@tempswatrue
          \fi
        \fi
```

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

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

```
      \if@tempswa
        \ifx\um@char@num@range\@empty
          \g@addto@macro\um@char@num@range{#1}
        \else
          \g@addto@macro\um@char@num@range{,#1}
        \fi
        #4%
      \fi
    \fi
  }
}
\def\um@firstof#1#2\@nil{#1}
\edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
\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`.

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

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

```
  \expandafter\ifx\expandafter\@marker\@tempb\relax
    \ifnum#4=#1\relax
      \@tempswatrue
    \fi
  \else
```

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

```
    \ifx\@empty\@tempb
      \ifnum#4>\numexpr#1-1\relax
        \@tempswatrue
      \fi
    \else
```

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

```
      \ifx\@empty\@tempa
        \ifnum#4<\numexpr#2+1\relax
          \@tempswatrue
        \fi
```

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

```
      \else
        \ifnum#4>\numexpr#1-1\relax
          \ifnum#4<\numexpr#2+1\relax
            \@tempswatrue
          \fi
        \fi
      \fi
    \fi
  \fi
}
```

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

```
\cs_set:Nn \um_map_chars_range:nnn {
  \clist_map_variable:nNn {#2} \l_um_input_num {
    \prg_stepwise_variable:nnnNn{0}{1}{#1} \l_um_incr_num {
      \um_set_mathcode:nnnn
        {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
        {\mathalpha}{\um_symfont_tl}
        {\numexpr \l_um_incr_num + #3 \relax}
    }
  }
}
\cs_set:Nn \um_map_chars_latin:nn {
  \um_map_chars_range:nnn {25}{#1}{#2}
}
\cs_set:Nn \um_map_chars_greek:nn {
  \um_map_chars_range:nnn {24}{#1}{#2}
}
\cs_set:Nn \um_map_chars_numbers:nn {
  \um_map_chars_range:nnn {9}{#1}{#2}
}
\cs_set:Nn \um_map_char:nn {
  \um_map_chars_range:nnn {0}{#1}{#2}
}
```

`\um_set_mathalphabet_char:Nnnn` `#1` : Maths alphabet
`#2` : Input char(s)
`#3` : Output char
Loops through character ranges setting `\mathcode`.

```
\cs_set:Npn \exp_args:Nnff {\::n\::f\::f\:::}
\cs_new:Nn \um_set_mathalphabet_char:Nnn {
  \clist_map_variable:nNn {#2} \l_um_input_num {
    \exp_args:Nnff \um_mathmap:Nnn {#1}
```

```
      {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
  }
}
```

`\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`.

```
\cs_new:Nn \um_set_mathalph_range:nNnn {
  \clist_map_variable:nNn {#3} \l_um_input_num {
    \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
      \exp_args:Nnff \um_mathmap:Nnn {#2}
        {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
        {\number\numexpr \l_um_inc_num + #4 \relax}
    }
  }
}
\cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
  \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
}
\cs_new:Nn \um_set_mathalphabet_latin:Nnn {
  \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
}
\cs_new:Nn \um_set_mathalphabet_greek:Nnn {
  \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
}
```

BCDBCDEABCDEFG

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

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.

```
\AtBeginDocument{\um@resolve@greek}
\newcommand\um@resolve@greek{
  \def\Alpha{\mitAlpha}
  \def\Beta{\mitBeta}
  \def\Gamma{\mitGamma}
  \def\Delta{\mitDelta}
  \def\Epsilon{\mitEpsilon}
```

```
\def\Zeta{\mitZeta}
\def\Eta{\mitEta}
\def\Theta{\mitTheta}
\def\Iota{\mitIota}
\def\Kappa{\mitKappa}
\def\Lambda{\mitLambda}
\def\Mu{\mitMu}
\def\Nu{\mitNu}
\def\Xi{\mitXi}
\def\Omicron{\mitOmicron}
\def\Pi{\mitPi}
\def\Rho{\mitRho}
\def\varTheta{\mitvarTheta}
\def\Sigma{\mitSigma}
\def\Tau{\mitTau}
\def\Upsilon{\mitUpsilon}
\def\Phi{\mitPhi}
\def\Chi{\mitChi}
\def\Psi{\mitPsi}
\def\Omega{\mitOmega}
```

Lowercase:

```
\def\alpha{\mitalpha}
\def\beta{\mitbeta}
\def\gamma{\mitgamma}
\def\delta{\mitdelta}
\def\epsilon{
  \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
}
\def\zeta{\mitzeta}
\def\eta{\miteta}
\def\theta{\mittheta}
\def\iota{\mitiota}
\def\kappa{\mitkappa}
\def\lambda{\mitlambda}
\def\mu{\mitmu}
\def\nu{\mitnu}
\def\xi{\mitxi}
\def\omicron{\mitomicron}
\def\pi{\mitpi}
\def\rho{\mitrho}
\def\varsigma{\mitvarsigma}
\def\sigma{\mitsigma}
\def\tau{\mittau}
\def\upsilon{\mitupsilon}
\def\phi{
  \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}
```

```
  }
  \def\chi{\mitchi}
  \def\psi{\mitpsi}
  \def\omega{\mitomega}
  \def\varepsilon{
      \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
  }
  \def\vartheta{\mitvartheta}
  \def\varkappa{\mitvarkappa}
  \def\varphi{
    \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
  }
  \def\varrho{\mitvarrho}
  \def\varpi{\mitvarpi}
}
```

6.5 Setting up the mappings

`\um_setup_literals:` : TODO : other literal symbols

```
\cs_set:Nn \um_setup_literals: {
  \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}
  \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}
  \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}
  \um_map_char:nn {\um@usv@ith}{\um@usv@ith}
  \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}
  \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}
  \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}
  \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}
  \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}
}
```

`\um_setup_bf_literals:` TODO: other literal symbols

```
\cs_set:Nn \um_setup_bf_literals: {
  \um_map_chars_latin:nn {\um@usv@bfupLatin}{\um@usv@bfupLatin}
  \um_map_chars_latin:nn {\um@usv@bfuplatin}{\um@usv@bfuplatin}
  \um_map_chars_latin:nn {\um@usv@bfitLatin}{\um@usv@bfitLatin}
  \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}
  \um_map_chars_greek:nn {\um@usv@bfupGreek}{\um@usv@bfupGreek}
  \um_map_chars_greek:nn {\um@usv@bfupgreek}{\um@usv@bfupgreek}
  \um_map_chars_greek:nn {\um@usv@bfitGreek}{\um@usv@bfitGreek}
  \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
}
```

`\um_setup_Latin:`

```
\cs_set:Nn \um_setup_Latin: {
  \if@um@upLatin
```

```
    \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
  \else
    \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
  \fi
}
```

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

```
\cs_set:Nn \um_setup_latin: {
  \if@um@uplatin
    \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
      \um_map_char:nn {\um@usv@ith}{`\h}
  \else
    \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
      \um_map_char:nn {`\h,\um@usv@ith}{\um@usv@ith}
  \fi
}
```

`\um_setup_Greek:`

```
\cs_set:Nn \um_setup_Greek: {
  \if@um@upGreek
    \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
      \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
  \else
    \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
      \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}
  \fi
}
```

`\um_setup_greek:`

```
\cs_set:Nn \um_setup_greek: {
  \if@um@upgreek
    \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
    \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
    \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
    \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
      \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
      \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
      \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
  \else
    \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
    \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
    \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
    \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
      \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
      \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
```

```
    \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
  \fi
}
```

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.

```
\cs_new:Nn \um_setup_alphabets: {
  \um_setup_math_alphabet:nn {up    }{latin,Latin,greek,Greek}
  \um_setup_math_alphabet:nn {it    }{latin,Latin,greek,Greek}
  \um_setup_math_alphabet:nn {bb    }{latin,Latin,num}
  \um_setup_math_alphabet:nn {scr   }{latin,Latin}
  \um_setup_math_alphabet:nn {frak  }{latin,Latin}
  \um_setup_math_alphabet:nn {sf    }{latin,Latin,num}
  \um_setup_math_alphabet:nn {sfup  }{latin,Latin,num}
  \um_setup_math_alphabet:nn {sfit  }{latin,Latin,num}
  \um_setup_math_alphabet:nn {tt    }{latin,Latin,num}
  \um_setup_math_alphabet:nn {bf    }{latin,Latin,greek,Greek,num}
  \um_setup_math_alphabet:nn {bfup  }{latin,Latin,greek,Greek,num}
  \um_setup_math_alphabet:nn {bfit  }{latin,Latin,greek,Greek,num}
  \um_setup_math_alphabet:nn {bfscr }{latin,Latin}
  \um_setup_math_alphabet:nn {bffrak}{latin,Latin}
```

```
  \um_setup_math_alphabet:nn {bfsf  }{latin,Latin,greek,Greek,num}
  \um_setup_math_alphabet:nn {bfsfup}{latin,Latin,greek,Greek,num}
  \um_setup_math_alphabet:nn {bfsfit}{latin,Latin,greek,Greek,num}
}
```

`\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.

```
\cs_new:Nn \um_setup_math_alphabet:nn {
  \clist_map_inline:nn {#2} {
    \um_glyph_if_exist:nT {\csname um@usv@#1##1 \endcsname}{
      \um_maybe_init_alphabet:n {#1}
      \um_prepare_alph:n {#1}
      \clist_map_break:
    }
  }
  \clist_map_inline:nn {#2} {
    \um_glyph_if_exist:nTF {\csname um@usv@#1##1 \endcsname}{
      \use:c {um_config_math#1_##1:}
    }{
      \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
      Math~ alphabet~
      \@backslashchar math#1~
      (\tl_use:c{g_um_math_alphabet_name_##1_tl})~
      not~ found~ in~ font~
      \fontname\um@font}
    }
  }
}
\tl_set:Nn \g_um_math_alphabet_name_latin_tl {Latin, lowercase}
\tl_set:Nn \g_um_math_alphabet_name_Latin_tl {Latin, uppercase}
\tl_set:Nn \g_um_math_alphabet_name_greek_tl {Greek, lowercase}
\tl_set:Nn \g_um_math_alphabet_name_Greek_tl {Greek, uppercase}
\tl_set:Nn \g_um_math_alphabet_name_num_tl   {Numerals}

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

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

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

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

```
\cs_new:Nn \um_prepare_alph:n {
  \cs_if_exist:cF {um_math#1:n} {
    \cs_set:cpn {um_math#1:n} ##1 {
      \use:c {um_setup_math#1:} ##1 \egroup
    }
    \cs_set_protected:cpn {math#1} {
      \bgroup
      \mode_if_math:F {
        \egroup\expandafter
        \non@alpherr\expandafter{\csname math#1\endcsname\space}
      }
      \use:c {um_math#1:n}
    }
  }
}
```

: 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.

```
\cs_new:Npn \um_config_mathup_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
}
\cs_new:Npn \um_config_mathup_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@ith} {`\h}
}
\cs_new:Npn \um_config_mathup_Greek: {
  \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varThe
}
\cs_new:Npn \um_config_mathup_greek: {
  \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@va
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@varthe
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkap
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
  \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
```

```
}
```

7.1.2 Italic: \mathit

```
\cs_new:Npn \um_config_mathit_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
}
\cs_new:Npn \um_config_mathit_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
   \um_set_mathalphabet_char:Nnn{\mathit}{`\h,\um@usv@ith}{\um@usv@ith}
}
\cs_new:Npn \um_config_mathit_Greek: {
  \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarT
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
}
\cs_new:Npn \um_config_mathit_greek: {
  \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartia
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@it
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvart
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvark
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
}
```

7.1.3 Blackboard or double-struck: \mathbb

```
\cs_new:Npn \um_config_mathbb_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}
   \um_set_mathalphabet_char:Nnn{\mathbb}{\um@usv@ith} {"1D559}
}
\cs_new:Npn \um_config_mathbb_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
   \um_set_mathalphabet_char:Nnn{\mathbb}{`\C,"1D436}{"2102}
   \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D43B}{"210D}
   \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D441}{"2115}
   \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D443}{"2119}
   \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D444}{"211A}
   \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D445}{"211D}
   \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D44D} {"2124}
}
\cs_new:Npn \um_config_mathbb_num: {
   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@num}{\um@usv@bbnum}
}
```

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

```
\cs_new:Npn \um_config_mathscr_Latin: {
  \um_set_mathalphabet_latin:Nnn  \mathscr {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin
   \um_set_mathalphabet_char:Nnn  \mathscr {`\B,"1D435}{"212C}
   \um_set_mathalphabet_char:Nnn  \mathscr {`\E,"1D438}{"2130}
   \um_set_mathalphabet_char:Nnn  \mathscr {`\F,"1D439}{"2131}
   \um_set_mathalphabet_char:Nnn  \mathscr {`\H,"1D43B}{"210B}
   \um_set_mathalphabet_char:Nnn  \mathscr {`\I,"1D43C}{"2110}
   \um_set_mathalphabet_char:Nnn  \mathscr {`\L,"1D43F}{"2112}
   \um_set_mathalphabet_char:Nnn  \mathscr {`\M,"1D440}{"2133}
   \um_set_mathalphabet_char:Nnn  \mathscr {`\R,"1D445}{"211B}
}
\cs_new:Npn \um_config_mathscr_latin: {
  \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrlatin}
   \um_set_mathalphabet_char:Nnn \mathscr {`\e,"1D452}{"212F}
   \um_set_mathalphabet_char:Nnn \mathscr {`\g,"1D454}{"210A}
   \um_set_mathalphabet_char:Nnn \mathscr {`\o,"1D45C}{"2134}
   \um_set_mathalphabet_char:Nnn \mathscr {\um@usv@ith} {"1D4BD}
}
```

7.1.5 Fractur or fraktur or blackletter: `\mathfrak`

```
\cs_new:Npn \um_config_mathfrak_Latin: {
  \um_set_mathalphabet_latin:Nnn  \mathfrak {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLat
   \um_set_mathalphabet_char:Nnn  \mathfrak {`\C,"1D436}{"212D}
   \um_set_mathalphabet_char:Nnn  \mathfrak {`\H,"1D43B}{"210C}
   \um_set_mathalphabet_char:Nnn  \mathfrak {`\I,"1D43C}{"2111}
   \um_set_mathalphabet_char:Nnn  \mathfrak {`\R,"1D445}{"211C}
   \um_set_mathalphabet_char:Nnn  \mathfrak {`\Z,"1D44D}{"2128}
}
\cs_new:Npn \um_config_mathfrak_latin: {
  \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklati
   \um_set_mathalphabet_char:Nnn \mathfrak {\um@usv@ith} {"1D525}
}
```

7.1.6 Sans serif: `\mathsf`

```
\cs_new:Npn \um_config_mathsf_Latin: {
   \bool_if:NTF \g_um_sfliteral_bool {
    \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin}{\um@usv@sfupLatin}
    \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itLatin}{\um@usv@sfitLatin}
   }{
    \um_set_mathalphabet_latin:Nnn  \mathsf {\um@usv@upLatin,\um@usv@itLatin}{ \um_sf_Latin_up
   }
}
\cs_new:Npn \um_config_mathsf_latin: {
   \bool_if:NTF \g_um_sfliteral_bool {
    \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin}{\um@usv@sfuplatin}
```

```
    \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itlatin}{\um@usv@sfitlatin}
      \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}
    }{
     \um_set_mathalphabet_latin:Nnn  \mathsf {\um@usv@uplatin,\um@usv@itlatin}{ \um_sf_latin_up
      \bool_if:NTF \g_um_upsans_bool {
        \um_set_mathalphabet_char:Nnn   \mathsf {\um@usv@ith} {"1D5C1}
      }{
        \um_set_mathalphabet_char:Nnn   \mathsf {\um@usv@ith} {"1D629}
      }
    }
}
\cs_new:Npn \um_config_mathsf_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnum}
}
```

7.1.7 Sans serif upright: `\mathsfup`

```
\cs_new:Npn \um_config_mathsfup_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathsfup}{\um@usv@num}{\um@usv@sfnum}
}
\cs_new:Npn \um_config_mathsfup_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfuplat
  \um_set_mathalphabet_char:Nnn \mathsfup {\um@usv@ith} {"1D5C1}
}
\cs_new:Npn \um_config_mathsfup_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfupLat
}
```

7.1.8 Sans serif italic: `\mathsfit`

Map the numbers like that because it seems sensible.

```
\cs_new:Npn \um_config_mathsfit_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathsfit}{\um@usv@num}{\um@usv@sfnum}
}
\cs_new:Npn \um_config_mathsfit_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLat
}
\cs_new:Npn \um_config_mathsfit_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlat
  \um_set_mathalphabet_char:Nnn \mathsfit {\um@usv@ith} {"1D629}
}
```

7.1.9 Typewriter or monospaced: `\mathtt`

```
\cs_new:Npn \um_config_mathtt_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
}
\cs_new:Npn \um_config_mathtt_Latin: {
```

```
  \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
}
\cs_new:Npn \um_config_mathtt_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlatin}
  \um_set_mathalphabet_char:Nnn \mathtt {\um@usv@ith} {"1D691}
}
```

7.2 Bold math alphabets

7.2.1 Bold: `\mathbf`

```
\cs_new:Npn \um_config_mathbf_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
}
\cs_new:Npn \um_config_mathbf_Latin: {
  \if@um@bfliteral
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfupLatin}
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
  \else
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um_bf_Latin_up_
  \fi
}
\cs_new:Npn \um_config_mathbf_latin: {
  \if@um@bfliteral
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@bfuplatin}
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@bfitlatin}
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
  \else
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um_bf_latin_up_
    \if@um@bfuplatin
      \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
    \else
      \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
    \fi
  \fi
}
\cs_new:Npn \um_config_mathbf_Greek: {
  \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
  \if@um@bfliteral
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfupGreek}
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
  \else
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um_bf_Greek_up_
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um_bfNabla_up_or_i
```

```
    \if@um@bfupGreek
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfva
    \else
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfit
    \fi
  \fi
}
\cs_new:Npn \um_config_mathbf_greek: {
  \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
  \if@um@bfliteral
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfupgreek}
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitgreek}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bfvarepsilon}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
  \else
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um_bf_greek_up_
    \if@um@bfupgreek
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfva
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfva
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
    \else
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfit
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfit
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarp
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarr
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
    \fi
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um_bfpartial_u
  \fi
}
```

7.2.2 Bold Italic: `\mathbfit`

```
\cs_new:Npn \um_config_mathbfit_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnum}
}
\cs_new:Npn \um_config_mathbfit_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLat
}
\cs_new:Npn \um_config_mathbfit_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlat
   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@ith} {"1D489}
}
\cs_new:Npn \um_config_mathbfit_Greek: {
  \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGre
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfit
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}
}
\cs_new:Npn \um_config_mathbfit_greek: {
  \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLat
  \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgre
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpa
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfit
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfit
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarp
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarr
  \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
}
```

7.2.3 Bold Italic: `\mathbfup`

```
\cs_new:Npn \um_config_mathbfup_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnum}
}
\cs_new:Npn \um_config_mathbfup_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLat
}
\cs_new:Npn \um_config_mathbfup_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplat
   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@ith} {"1D421}
}
\cs_new:Npn \um_config_mathbfup_Greek: {
  \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGre
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfva
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Digamma}{\um@usv@bfDigamma}
}
\cs_new:Npn \um_config_mathbfup_greek: {
```

```
  \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgre
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpart
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfva
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfva
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
  \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@digamma}{\um@usv@bfdigamma}
}
```

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

```
\cs_new:Npn \um_config_mathbffrak_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin, \um@usv@itLatin}{\um@usv@bffra
}
\cs_new:Npn \um_config_mathbffrak_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bffra
   \um_set_mathalphabet_char:Nnn{\mathbffrak}{\um@usv@ith} {"1D58D}
}
```

7.2.5 Bold script or calligraphic: `\mathbfscr`

```
\cs_new:Npn \um_config_mathbfscr_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrL
}
\cs_new:Npn \um_config_mathbfscr_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrl
   \um_set_mathalphabet_char:Nnn{\mathbfscr}{\um@usv@ith} {"1D4F1}
}
```

7.2.6 Bold sans serif: `\mathbfsf`

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

```
\cs_new:Npn \um_config_mathbfsf_num: {
  \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
}
\cs_new:Npn \um_config_mathbfsf_Latin: {
  \bool_if:NTF \g_um_sfliteral_bool {
   \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@upLatin}{\um@usv@bfsfupLatin}
   \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@itLatin}{\um@usv@bfsfitLatin}
  }{
   \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um_bfsf_Lati
  }
}
\cs_new:Npn \um_config_mathbfsf_latin: {
  \bool_if:NTF \g_um_sfliteral_bool {
   \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@uplatin}{\um@usv@bfsfuplatin}
```

```
    \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@itlatin}{\um@usv@bfsfitlatin}
      \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
    }{
    \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um_bfsf_lati
      \bool_if:NTF \g_um_upsans_bool {
        \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D5F5}
      }{
        \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
      }
    }
}
\cs_new:Npn \um_config_mathbfsf_Greek: {
    \bool_if:NTF \g_um_sfliteral_bool {
    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@upGreek}{\um@usv@bfsfupGreek}
    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@itGreek}{\um@usv@bfsfitGreek}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varTheta}{"1D767}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varTheta}{"1D7A1}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@Nabla}{\um@usv@bfsfNabla}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@itNabla}{\um@usv@bfsfitNabla}
    }{
    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um_bfsf_Gree
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{\um_bfsfNabla_up
      \bool_if:NTF \g_um_upsans_bool {
      \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{"1D767}
      }{
      \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{"1D7A1}
      }
    }
}
\cs_new:Npn \um_config_mathbfsf_greek: {
    \bool_if:NTF \g_um_sfliteral_bool {
    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@upgreek}{\um@usv@bfsfupgreek}
    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@itgreek}{\um@usv@bfsfitgreek}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@partial}{\um@usv@bfsfpartial}
     \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varepsilon}{"1D78A}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@vartheta}{"1D78B}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varkappa}{"1D78C}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varphi}{"1D78D}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varrho}{"1D78E}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varpi}{"1D78F}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@itpartial}{\um@usv@bfsfitpartial}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@itvarepsilon}{"1D7C4}
     \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@itvartheta}{"1D7C5}
     \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@itvarkappa}{"1D7C6}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvarphi}{"1D7C7}
      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvarrho}{"1D7C8}
```

```
    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvarpi}{"1D7C9}
   }{
    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um_bfsf_gree
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@partial,\um@usv@itpartial}{\um_bfsfpart
    \bool_if:NTF \g_um_upsans_bool {
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D7
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{"1D78B}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{"1D78C}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{"1D78D}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{"1D78E}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{"1D78F}
    }{
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D7
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{"1D7C5}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{"1D7C6}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{"1D7C7}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{"1D7C8}
    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{"1D7C9}
    }
  }
}
```

7.2.7 Bold upright sans serif: `\mathbfsfup`

```
\cs_new:Npn \um_config_mathbfsfup_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfsfup}{\um@usv@num}{\um@usv@bfsfnum}
}
\cs_new:Npn \um_config_mathbfsfup_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfu
}
\cs_new:Npn \um_config_mathbfsfup_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfu
   \um_set_mathalphabet_char:Nnn \mathbfsfup {\um@usv@ith} {"1D5F5}
}
\cs_new:Npn \um_config_mathbfsfup_Greek: {
  \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfu
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varTheta,\um@usv@itvarTheta}{"1D767}
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@Nabla,\um@usv@itNabla}{"1D76F}
}
\cs_new:Npn \um_config_mathbfsfup_greek: {
  \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfu
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@partial,\um@usv@itpartial}{"1D789}
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D78A
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@vartheta,\um@usv@itvartheta}{"1D78B}
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varkappa,\um@usv@itvarkappa}{"1D78C}
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varphi,\um@usv@itvarphi}{"1D78D}
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varrho,\um@usv@itvarrho}{"1D78E}
```

```
  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varpi,\um@usv@itvarpi}{"1D78F}
}
```

7.2.8 Bold italic sans serif: `\mathbfsfit`

```
\cs_new:Npn \um_config_mathbfsfit_num: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@num}{\um@usv@bfsfnum}
}
\cs_new:Npn \um_config_mathbfsfit_Latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfi
}
\cs_new:Npn \um_config_mathbfsfit_latin: {
  \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfi
   \um_set_mathalphabet_char:Nnn \mathbfsfit {\um@usv@ith} {"1D65D}
}
\cs_new:Npn \um_config_mathbfsfit_Greek: {
  \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfi
   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{"1D7A1}
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNa
}
\cs_new:Npn \um_config_mathbfsfit_greek: {
  \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfi
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsf
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D7C4
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{"1D7C5}
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{"1D7C6}
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{"1D7C7}
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{"1D7C8}
  \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{"1D7C9}
}
```

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`
`\um@scanactivedef`

We need to do some trickery to transform the `\UnicodeMathSymbol` argument `"ABCDEF` into the XeTeX '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.

```
\begingroup
  \char_make_other:N \^
  \cs_gset:Npn \um@scancharlet#1="#2\@nil {
    \lowercase{
```

```
      \scantokens{\global\let#1=^^^^^#2}
    }
  }
```

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

```
  \gdef\um@scanactivedef"#1\@nil#2{
    \lowercase{
      \tl_rescan:nn{
        \ExplSyntaxOn
        \char_make_math_superscript:N\^
      }{
        \global\def^^^^^#1{#2}
      }
    }
  }
\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`.

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

Fix `\backslash`:

```
\group_begin:
  \lccode`\*=`\\
  \char_make_escape:N \|
  \char_make_other:N \\
  |lowercase{
|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.

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

\cs_new:Nn \um_nprimes:n {
  ^{
     \primesingle
    \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
    }
}
\cs_new:Nn \um_nprimes_select:n {
  \prg_case_int:nnn {#1}{
```

```
    {1} { ^{\primesingle} }
    {2} {
    \um_glyph_if_exist:nTF {"2033} { ^{\primedouble} } {\um_nprimes:n {#1}}
    }
    {3} {
    \um_glyph_if_exist:nTF {"2034} {^{\primetriple} } {\um_nprimes:n {#1}}
    }
    {4} {
    \um_glyph_if_exist:nTF {"2057} { ^{\primequadruple} } {\um_nprimes:n {#1}}
    }
  }{
    \um_nprimes:n {#1}
  }
}
```

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.

```
\cs_new:Nn \um_scanprime: {
  \num_zero:N \l_um_primecount_num
  \um_scanprime_collect:
}
\cs_new:Nn \um_scanprime_collect: {
  \num_incr:N \l_um_primecount_num
  \peek_meaning_remove:NTF ' {
    \um_scanprime_collect:
  }{
    \peek_meaning_remove:NTF \um_scanprime: {
      \um_scanprime_collect:
    }{
      \peek_meaning_remove:NTF ^^^^2032 {
        \um_scanprime_collect:
      }{
        \um_nprimes_select:n {\l_um_primecount_num}
      }
    }
  }
}

\cs_set_eq:NN \prime \um_scanprime:
\group_begin:
  \char_make_active:N \'
  \char_make_active:n {"2032}
  \cs_gset_eq:NN ' \um_scanprime:
  \cs_gset_eq:NN ^^^^2032 \um_scanprime:
\group_end:
```

8.0.2 Unicode radicals

Undo the damage made to `\sqrt`:

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

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

```
\def\r@@t#1#2{
  \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
  \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
  \raise \dimexpr(
      \um@fontdimen@percent{65}{\um@font}\ht\z@-
      \um@fontdimen@percent{65}{\um@font}\dp\z@
    )\relax
    \copy \rootbox
  \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
  \box \z@
}
```

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 XeTeX to typeset the results; this means that the actual subscript/superscript glyphs are never seen in the output document — they are only used as input characters.

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

First, the setup of each mathactive char:

```
\prop_new:N \g_um_supers_prop
\prop_new:N \g_um_subs_prop
\cs_generate_variant:Nn \prop_gput:Nnn {Nxn}
\cs_generate_variant:Nn \prop_get:NnN {cxN}
\cs_generate_variant:Nn \prop_if_in:NnTF {cx}

\group_begin:

% Populate a property list with superscript characters; their meaning as their key,
% for reasons that will become apparent soon, and their replacement as each key's value.
% Then make the superscript active and bind it to the scanning function.
```

```
%
% \cs{scantokens} makes this process much simpler since we can activate the char
% and assign its meaning in one step.
\cs_set:Nn \um_setup_active_superscript:nn {
  \prop_gput:Nxn \g_um_supers_prop   {\meaning #1} {#2}
  \char_make_active:n {`#1}
  \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
  \scantokens{
    \cs_gset:Npn #1 {
      \tl_set:Nn \l_um_ss_chain_tl {#2}
      \cs_set_eq:NN \um_sub_or_super:n \sp
      \tl_set:Nn \l_um_tmpa_tl {supers}
      \um_scan_sscript:
    }
  }
}

\um_setup_active_superscript:nn {^^^^2070} {0}
\um_setup_active_superscript:nn {^^^^00b9} {1}
\um_setup_active_superscript:nn {^^^^00b2} {2}
\um_setup_active_superscript:nn {^^^^00b3} {3}
\um_setup_active_superscript:nn {^^^^2074} {4}
\um_setup_active_superscript:nn {^^^^2075} {5}
\um_setup_active_superscript:nn {^^^^2076} {6}
\um_setup_active_superscript:nn {^^^^2077} {7}
\um_setup_active_superscript:nn {^^^^2078} {8}
\um_setup_active_superscript:nn {^^^^2079} {9}
\um_setup_active_superscript:nn {^^^^207a} {+}
\um_setup_active_superscript:nn {^^^^207b} {-}
\um_setup_active_superscript:nn {^^^^207c} {=}
\um_setup_active_superscript:nn {^^^^207d} {(}
\um_setup_active_superscript:nn {^^^^207e} {)}
\um_setup_active_superscript:nn {^^^^2071} {i}
\um_setup_active_superscript:nn {^^^^207f} {n}

% Ditto above.
\cs_set:Nn \um_setup_active_subscript:nn {
  \prop_gput:Nxn \g_um_subs_prop   {\meaning #1} {#2}
  \char_make_active:n {`#1}
  \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
  \scantokens{
    \cs_gset:Npn #1 {
      \tl_set:Nn \l_um_ss_chain_tl {#2}
      \cs_set_eq:NN \um_sub_or_super:n \sb
      \tl_set:Nn \l_um_tmpa_tl {subs}
```

```
      \um_scan_sscript:
    }
  }
}

\um_setup_active_subscript:nn {^^^^2080} {0}
\um_setup_active_subscript:nn {^^^^2081} {1}
\um_setup_active_subscript:nn {^^^^2082} {2}
\um_setup_active_subscript:nn {^^^^2083} {3}
\um_setup_active_subscript:nn {^^^^2084} {4}
\um_setup_active_subscript:nn {^^^^2085} {5}
\um_setup_active_subscript:nn {^^^^2086} {6}
\um_setup_active_subscript:nn {^^^^2087} {7}
\um_setup_active_subscript:nn {^^^^2088} {8}
\um_setup_active_subscript:nn {^^^^2089} {9}
\um_setup_active_subscript:nn {^^^^208a} {+}
\um_setup_active_subscript:nn {^^^^208b} {-}
\um_setup_active_subscript:nn {^^^^208c} {=}
\um_setup_active_subscript:nn {^^^^208d} {(}
\um_setup_active_subscript:nn {^^^^208e} {)}
\um_setup_active_subscript:nn {^^^^2090} {a}
\um_setup_active_subscript:nn {^^^^2091} {e}
\um_setup_active_subscript:nn {^^^^1d62} {i}
\um_setup_active_subscript:nn {^^^^2092} {o}
\um_setup_active_subscript:nn {^^^^1d63} {r}
\um_setup_active_subscript:nn {^^^^1d64} {u}
\um_setup_active_subscript:nn {^^^^1d65} {v}
\um_setup_active_subscript:nn {^^^^2093} {x}
\um_setup_active_subscript:nn {^^^^1d66} {\beta}
\um_setup_active_subscript:nn {^^^^1d67} {\gamma}
\um_setup_active_subscript:nn {^^^^1d68} {\rho}
\um_setup_active_subscript:nn {^^^^1d69} {\phi}
\um_setup_active_subscript:nn {^^^^1d6a} {\chi}

\group_end:

% The scanning command, evident in its purpose:
\cs_new:Nn \um_scan_sscript: {
  \um_scan_sscript:TF {
    \um_scan_sscript:
  }{
    \um_sub_or_super:n {\l_um_ss_chain_tl}
  }
}

% The main theme here is stolen from the source to the vari-
```

```
ous \cs{peek_} functions.
% Consider this function as simply boilerplate:
\cs_new:Nn \um_scan_sscript:TF {
  \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
  \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
  \tl_set:Nx \l_peek_false_tl {\exp_not:n{\group_align_safe_end: #2}}
  \group_align_safe_begin:
    \peek_after:NN \um_peek_execute_branches_ss:
}

% We do not skip spaces when scanning ahead, and we explicitly wish to
% bail out on encountering a space or a brace.
\cs_new:Npn \um_peek_execute_branches_ss: {
  \bool_if:nTF {
    \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
    \token_if_eq_catcode_p:NN \l_peek_token \c_group_end_token ||
    \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
  }
  { \l_peek_false_tl  }
  { \um_peek_execute_branches_ss_aux: }
}

% This is the actual comparison code.
% Because the peeking has already tokenised the next token,
% it's too late to extract its charcode directly. Instead,
% we look at its meaning, which remains a `character' even
% though it is itself math-active. If the character is ever
% made fully active, this will break our assumptions!
%
% If the char's meaning exists as a property list key, we
% build up a chain of sub-/superscripts and iterate. (If not, exit and
% typeset what we've already collected.)
\cs_new:Nn \um_peek_execute_branches_ss_aux: {
  \prop_if_in:cxTF
    {g_um_\l_um_tmpa_tl _prop}
    {\meaning\l_peek_token}
    {
      \prop_get:cxN
        {g_um_\l_um_tmpa_tl _prop}
        {\meaning\l_peek_token}
        \l_um_tmpb_tl
      \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmpb_tl
      \l_peek_true_tl
    }
    {\l_peek_false_tl}
}
```

8.0.4 Synonyms and all the rest

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

```
\def\operator@font{\um_setup_mathup:}
\def\to{\rightarrow}
\def\vec{\overrightarrow}
\def\le{\leq}
\def\ge{\geq}
\def\neq{\ne}
```

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

```
\@ifpackageloaded{amsmath}{
  % define their own colon, perhaps I should just steal it.
}{
  \cs_set_protected:Npn \colon {
    \bool_if:NTF \g_um_literal_colon_bool {:} { \mathpunct{:} }
  }
}
```

`\mathcal`

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

`\mathrm`

```
\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:

```
  \@ifpackageloaded{amsmath}{
    \tl_remove_in:Nn \@begindocumenthook {
      \mathchardef\std@minus\mathcode`\-\relax
      \mathchardef\std@equal\mathcode`\=\relax
    }
  }{}
```

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

```
  \@ifpackageloaded{amsopn}{
```

```
        \cs_set:Npn \newmcodes@ {
          \mathcode`\'39
          \mathcode`\*42
          \mathcode`\."613A%
      %  \ifnum\mathcode`\-=45 \else
      %    \mathchardef\std@minus\mathcode`\-\relax
      %  \fi
          \mathcode`\-45
          \mathcode`\/47
          \mathcode`\:"603A\relax
        }
      }{}
```

Octothorpe is an odd one:

```
\AtBeginDocument{
  \def\#{\mode_if_math:TF{\mathoctothorpe}{\char`\#}}
  \def\widehat{\hat}
  \def\widetilde{\tilde}
}
```

\digamma
\Digamma

I might end up just changing these in the table.

```
\def\digamma{\updigamma}
\def\Digamma{\upDigamma}
```

Overriding amsmath definitions:

```
\AtBeginDocument{
  \def\@cdots{\mathinner{\cdots}}
}
```

Interaction with beamer:

```
\@ifclassloaded{beamer}{
  \ifbeamer@suppressreplacements\else
    \PackageWarningNoLine{unicode-math}{
      Disabling~ beamer's~ math~ setup.^^J
      Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
    }
    \beamer@suppressreplacementstrue
  \fi
}{}
```

The end.

```
\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.

```
#!/bin/sh

cat stix-tbl.txt |
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!)…

```
 {if (usv != substr($0,2,5) && substr($0,2,1) != " ")
    {usv = substr($0,2,5);
     texname = substr($0,84,25);
     class = substr($0,57,1);
     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):

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

Print the actual entry corresponding to the unicode character:

```
     print "\\UnicodeMathSymbol{\"" \
           usv "}{" \
           texname "}{" \
           class "}{" \
           description "}%";
    }}' - |
```

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

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

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

```
    -e ' s/{F}/{\\mathord}/   ' \
    -e ' s/{A}/{\\mathalpha}/ ' \
    -e ' s/{D}/{\\mathaccent}/ ' \
    -e ' s/{P}/{\\mathpunct}/ ' \
    -e ' s/{B}/{\\mathbin}/   ' \
    -e ' s/{R}/{\\mathrel}/   ' \
    -e ' s/{L}/{\\mathop}/    ' \
    -e ' s/{O}/{\\mathopen}/  ' \
    -e ' s/{C}/{\\mathclose}/ ' \
```

Fixing up a couple of things in the STIX table.

```
    -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: $\propto$, $\leq$, $\rightarrow$

`\DeclareSymbolFont{`⟨*name*⟩`}`⟨*NFSS decl.*⟩
Declares a named maths font such as `operators` from which symbols are defined with `\DeclareMathSymbol`.

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

`\DeclareMathAlphabet{`⟨*cmd*⟩`}`⟨*NFSS decl.*⟩

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

`\DeclareSymbolFontAlphabet{`⟨*cmd*⟩`}{`⟨*name*⟩`}`

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

Maths 'versions' Different maths weights can be defined with the following, switched in text with the `\mathversion{`⟨*maths version*⟩`}` command.

`\SetSymbolFont{`⟨*name*⟩`}{`⟨*maths version*⟩`}`⟨*NFSS decl.*⟩
`\SetMathAlphabet{`⟨*cmd*⟩`}{`⟨*maths version*⟩`}`⟨*NFSS decl.*⟩

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

Delimiters and radicals use wrappers around TEX's `\delimiter/\radical` primitives, which are re-designed in XETEX. The syntax used in LATEX's NFSS is therefore not so relevant here.

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

`\DeclareMathDelimiter{`⟨*symbol*⟩`}{`⟨*type*⟩`}{`⟨*sym. font*⟩`}{`⟨*slot*⟩`}{`⟨*sym. font*⟩`}{`⟨*slot*⟩`}`

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

In those cases, glyph slots in *two* symbol fonts are required; one for the small ('regular') case, the other for situations when the glyph is larger. This is not the case in XETEX.

Accents are not included yet.

Summary For symbols, something like:

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

For characters, something like:

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

File III

XETEX math font dimensions

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

\fontdimen	Dimension name	Description
10	ScriptPercentScaleDown	Percentage of scaling down for script level 1. Suggested value: 80%.
11	ScriptScriptPercentScaleDown	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	DelimitedSubFormulaMinHeight	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height × 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	FlattenedAccentBaseHeight	Maximum (ink) height of accent base that does not require flattening the accents. Suggested: cap height of the font (os2.sCapHeight).
18	SubscriptShiftDown	The standard shift down applied to subscript elements. Positive for moving in the downward direction. Suggested: os2.ySubscriptYOffset.
19	SubscriptTopMax	Maximum allowed height of the (ink) top of subscripts that does not require moving subscripts further down. Suggested: /5 x-height.

\fontdimen	Dimension name	Description
20	SubscriptBaselineDropMin	Minimum allowed drop of the baseline of subscripts relative to the (ink) bottom of the base. Checked for bases that are treated as a box or extended shape. Positive for subscript baseline dropped below the base bottom.
21	SuperscriptShiftUp	Standard shift up applied to superscript elements. Suggested: os2.ySuperscriptYOffset.
22	SuperscriptShiftUpCramped	Standard shift of superscripts relative to the base, in cramped style.
23	SuperscriptBottomMin	Minimum allowed height of the (ink) bottom of superscripts that does not require moving subscripts further up. Suggested: ¼ x-height.
24	SuperscriptBaselineDrop-Max	Maximum allowed drop of the baseline of superscripts relative to the (ink) top of the base. Checked for bases that are treated as a box or extended shape. Positive for superscript baseline below the base top.
25	SubSuperscriptGapMin	Minimum gap between the superscript and subscript ink. Suggested: 4×default rule thickness.
26	SuperscriptBottomMax-WithSubscript	The maximum level to which the (ink) bottom of superscript can be pushed to increase the gap between superscript and subscript, before subscript starts being moved down. Suggested: /5 x-height.
27	SpaceAfterScript	Extra white space to be added after each subscript and superscript. Suggested: 0.5pt for a 12 pt font.
28	UpperLimitGapMin	Minimum gap between the (ink) bottom of the upper limit, and the (ink) top of the base operator.
29	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	LowerLimitBaselineDropMin	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	FractionDenominatorShiftDown	Standard shift down applied to the denominator. Positive for moving in the downward direction.
45	FractionDenominatorDisplayStyleShiftDown	Standard shift down applied to the denominator in display style. Positive for moving in the downward direction. Suggested: StackBottomDisplayStyleShiftDown.
46	FractionNumeratorGapMin	Minimum tolerated gap between the (ink) bottom of the numerator and the ink of the fraction bar. Suggested: default rule thickness
47	FractionNumDisplayStyleGapMin	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	FractionDenominatorGapMin	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar. Suggested: default rule thickness
50	FractionDenomDisplayStyleGapMin	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	SkewedFractionHorizontalGap	Horizontal distance between the top and bottom elements of a skewed fraction.
52	SkewedFractionVerticalGap	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 he 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
\@begindocumenthook 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
\@tempswatrue 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 658

\addtoversion 312
\alloc@ 365
\Alpha 808
\alpha 833
\alpha@elt 316
\alpha@list 315
\AtBeginDocument 806, 1707, 1714
\awint 654

B

\B 1067
\beamer@suppressreplacementstrue 1723
\begingroup 371, 662, 1431, 1449
\Beta 809
\beta 834, 1610
\bgroup 995
\bool_if:NF 496
\bool_if:NTF 222, 838, 857, 863, 868, 1096, 1104, 1110, 1304, 1312, 1318, 1326, 1336, 1344, 1364, 1683
\bool_if:nTF 1640
\bool_new:N 22, 209, 210, 281
\bool_set_false:N 145, 146, 172, 177, 214, 276, 286
\bool_set_true:N 158, 159, 171, 176, 216, 218, 278, 284
\box 1524

C

\C 1054, 1085
\c_group_begin_token 1641
\c_group_end_token 1642
\c_peek_true_remove_next_tl . . . 1631
\c_space_token 1643
\cdots 1715
\cdp@elt 303
\cdp@list 302
\char 1708
\char_make_active:N 1509
\char_make_active:n 372, 1510, 1542, 1575
\char_make_escape:N 1458
\char_make_math_superscript:N . . 1442
\char_make_other:N . . 1432, 1453, 1459
\chardef 365
\Chi 830
\chi 859, 1614
\cirfnint 654
\clist_map_break: 962
\clist_map_inline:Nn 638
\clist_map_inline:nn 530, 633, 640, 958, 965
\clist_map_variable:NNn 703
\clist_map_variable:nNn . 760, 783, 789
\colon 1682
\copy 1522
\cs 1538, 1627
\cs_generate_variant:Nn . . . 1528–1530
\cs_gset:cpn 379, 389
\cs_gset:Npn . . 392, 397, 1433, 1545, 1578
\cs_gset:Npx 402
\cs_gset_eq:NN 1511, 1512
\cs_if_exist:cF 990
\cs_new:Nn 479, 493, 524, 529, 534, 537, 543, 574, 577, 782, 788, 797, 800, 803, 938, 957, 989, 1465, 1471, 1487, 1491, 1619, 1629, 1659
\cs_new:Npn 1004, 1007, 1011, 1016, 1026, 1029, 1033, 1038, 1048, 1052, 1062, 1065, 1076, 1083, 1091, 1095, 1103, 1117, 1120, 1123, 1127, 1130, 1133, 1136, 1140, 1143, 1146, 1150, 1153, 1161, 1175, 1194, 1233, 1236, 1239, 1243, 1248, 1259, 1262, 1265, 1269, 1275, 1286, 1289, 1293, 1296, 1300, 1303, 1311, 1325, 1343, 1381, 1384, 1387, 1391, 1396, 1406, 1409, 1412, 1416, 1421, 1639
\cs_set:cpn 991
\cs_set:Nn 203, 221, 245, 263, 411, 485, 488, 580, 632, 637, 759, 769, 772, 775, 778, 873, 884, 894, 901, 910, 919, 983, 1540, 1573
\cs_set:Npn 781, 1695
\cs_set_eq:cN 984
\cs_set_eq:NN 457–460, 464–467, 1507, 1547, 1580
\cs_set_protected:cpn 994
\cs_set_protected:Npn 1682
\cs_to_str:N 374, 379, 389, 647
\csname 368, 374, 377, 380, 412, 419, 539, 575, 578, 959, 966, 998

D

\DeclareDocumentCommand 414
\DeclareMathAccent 333
\DeclareMathAlphabet 329
\DeclareMathDelimiter 338
\DeclareMathRadical 344
\DeclareMathSizes 304
\DeclareMathSymbol 335
\DeclareMathVersion 309, 421
\DeclareRobustCommand 1514
\DeclareSymbolFont 322, 469
\DeclareSymbolFontAlphabet 346
\DeclareSymbolFontAlphabet@ 347
\def . . 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
\define@choicekey 132, 180, 211, 238, 256, 274, 282, 289, 695
\define@cmdkey 691–694
\define@key 674
\define@mathalphabet 310
\define@mathgroup 311
\Delta 811
\delta 836
\Digamma 1712
\digamma 1712
\dimexpr 352, 445, 1518
\do 349
\dorestore@version 319
\dp 1520

E

\E 1068
\e 1078
\edef 427, 730, 731
\egroup 992, 997
\else 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
\else: 987
\encodingdefault 470
\endcsname 368, 374, 377, 380, 412, 419, 539, 575, 578, 959, 966, 998
\endgroup 375, 668, 1448, 1455
\Epsilon 812
\epsilon 837
\Eta 814
\eta 841
\etex_iffontchar:D 987
\ExecuteOptionsX 298
\exp_args:Nnff 781, 784, 791
\exp_args:No 634, 641
\exp_not:n 1630, 1632
\expandafter 368, 376, 381, 383, 387, 659, 706, 708, 711, 716, 730, 731, 735, 997, 998
\ExplSyntaxOff 1726
\ExplSyntaxOn 6, 1441

F

\F 1069
\f@size 419
\fi 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
\fi: 987
\fint 654
\font 444
\fontdimen 352, 445, 1517, 1523
\fontname 974

G

\g 1079
\g@addto@macro 720, 722
\g_um_literal_colon_bool 281, 284, 286, 496, 1683
\g_um_math_alphabet_name_Greek_tl 981
\g_um_math_alphabet_name_greek_tl 980

\g_um_math_alphabet_name_Latin_tl 979
\g_um_math_alphabet_name_latin_tl 978
\g_um_math_alphabet_name_num_tl . 982
\g_um_primekern_muskip 1462, 1463, 1468
\g_um_sfliteral_bool . 176, 210, 218, 1096, 1104, 1304, 1312, 1326, 1344
\g_um_slash_delimiter_usv 291, 293, 295, 544–546
\g_um_subs_prop 1527, 1574
\g_um_supers_prop 1526, 1541
\g_um_texgreek_bool . 22, 146, 159, 172, 177, 276, 278, 838, 857, 863, 868
\g_um_upsans_bool 145, 158, 171, 209, 214, 216, 222, 1110, 1318, 1336, 1364
\Gamma 810
\gamma 835, 1611
\gdef 1438
\ge 1677
\geq 1677
\get@cdp 328
\glb@currsize 415
\global 373, 376, 393, 394, 398, 399, 404, 697, 1435, 1444, 1543, 1576
\group@elt 324
\group@list 323
\group_align_safe_begin: 1633
\group_align_safe_end: 1632
\group_begin: 1456, 1508, 1532
\group_end: 1513, 1616

H

\H 1055, 1070, 1086
\h 904, 907, 1009, 1031
\hat 1709
\hbox 1516
\ht 1519

I

\I 1071, 1087
\if@tempswa 718
\if@um@bfliteral 15, 508, 591, 1154, 1162, 1177, 1196
\if@um@bfupGreek 16, 204, 604, 1187
\if@um@bfupgreek 17, 205, 611, 1215
\if@um@bfupLatin 18, 206, 594
\if@um@bfuplatin 19, 207, 599, 1168
\if@um@fontspec@feature 8, 675
\if@um@literal 10, 499, 583
\if@um@ot@math@ 9
\if@um@upGreek 11, 911
\if@um@upgreek 12, 920
\if@um@upLatin 13, 895
\if@um@uplatin 14, 902
\if@um@upNabla 20, 246
\if@um@uppartial 21, 264
\ifbeamer@suppressreplacements . 1718
\ifcase 134, 181, 213, 239, 257, 275, 283, 290, 696
\ifdim 445
\ifin@ 382, 388
\ifnum . . 639, 736, 741, 746, 750, 751, 1699
\ifx 355, 358, 370, 391, 396, 401, 454, 581, 664, 704, 707, 708, 711, 719, 735, 740, 745
\iiiint 652
\iiint 652
\iint 652
\in@ 381, 387
\init@restore@version 318
\int 652
\intBar 654
\intbar 654
\intcap 656
\intclockwise 653
\intcup 656
\intlarhk 655
\intx 655
\Iota 816
\iota 843

K

\Kappa 817
\kappa 844
\kern 1517, 1523

L

\L 1072
\l_peek_false_tl 1632, 1645, 1671
\l_peek_token . . . 1641–1643, 1662, 1666
\l_peek_true_aux_tl 1630
\l_peek_true_tl 1631, 1669
\l_um_inc_num 790, 792, 793

\l_um_incr_num 761, 763, 765
\l_um_input_num 760, 763, 783, 785, 789, 792
\l_um_primecount_num 1464, 1488, 1492, 1502
\l_um_script_features_tl . . . 422, 433
\l_um_script_font_tl 424, 432
\l_um_ss_chain_tl 1546, 1579, 1623, 1668
\l_um_sscript_features_tl . . 423, 437
\l_um_sscript_font_tl 425, 436
\l_um_tmpa_tl . . . 1548, 1581, 1661, 1665
\l_um_tmpb_tl 1667, 1668
\Lambda 818
\lambda 845
\lccode 1457
\le 1676
\left 671
\left@primitive 671, 672
\leq 1676
\let 366, 415–417, 671, 697, 1435
\lowercase 1434, 1439
\lowint 656

M

\M 1073
\m@th 1516
\mathaccent 401
\mathalpha 648, 764
\mathbb 1049, 1050, 1053–1060, 1063
\mathbf 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
\mathbffrak 1287, 1290, 1291
\mathbfit 1234, 1237, 1240, 1241, 1244–1246, 1249–1257
\mathbfscr 1294, 1297, 1298
\mathbfsf 1301, 1305, 1306, 1308, 1313–1315, 1317, 1319, 1321, 1327–1332, 1334, 1335, 1337, 1339, 1345–1360, 1362, 1363, 1365–1370, 1372–1377
\mathbfsfit 1407, 1410, 1413, 1414, 1417–1419, 1422–1429
\mathbfsfup 1382, 1385, 1388, 1389, 1392–1394, 1397–1404
\mathbfup 1260, 1263, 1266, 1267, 1270–1273, 1276–1284
\mathbin 494, 495
\mathcal 1686
\mathchar@type 345, 378, 392, 394, 397, 399, 402, 404, 412, 538
\mathchardef 1690, 1691, 1700
\mathclose 396
\mathcode 373, 1690, 1691, 1696–1700, 1702–1704
\mathfrak 1084–1089, 1092, 1093
\mathgroup 365
\mathinner 1715
\mathit 1027, 1030, 1031, 1034–1036, 1039–1046
\mathoctothorpe 1708
\mathop 370
\mathopen 391, 672
\mathord 500–503, 505, 506, 509–516, 518–521, 535
\mathpunct 1683
\mathrel 497
\mathrm 1687
\mathscr . . . 1066–1074, 1077–1081, 1686
\mathsf 1097, 1098, 1100, 1105–1107, 1109, 1111, 1113, 1118
\mathsfit 1131, 1134, 1137, 1138
\mathsfup 1121, 1124, 1125, 1128
\mathtt 1141, 1144, 1147, 1148
\mathup 1005, 1008, 1009, 1012–1014, 1017–1024, 1687
\mddefault 470
\meaning 1541, 1574, 1662, 1666
\mitAlpha 808
\mitalpha 833
\mitBeta 809
\mitbeta 834
\mitChi 830
\mitchi 859
\mitDelta 811
\mitdelta 836
\mitEpsilon 812
\mitepsilon 838, 863
\mitEta 814
\miteta 841
\mitGamma 810

\mitgamma 835
\mitIota 816
\mitiota 843
\mitKappa 817
\mitkappa 844
\mitLambda 818
\mitlambda 845
\mitMu 819
\mitmu 846
\mitNu 820
\mitnu 847
\mitOmega 832
\mitomega 861
\mitOmicron 822
\mitomicron 849
\mitPhi 829
\mitphi 857, 868
\mitPi 823
\mitpi 850
\mitPsi 831
\mitpsi 860
\mitRho 824
\mitrho 851
\mitSigma 826
\mitsigma 853
\mitTau 827
\mittau 854
\mitTheta 815
\mittheta 842
\mitUpsilon 828
\mitupsilon 855
\mitvarepsilon 838, 863
\mitvarkappa 866
\mitvarphi 857, 868
\mitvarpi 871
\mitvarrho 870
\mitvarsigma 852
\mitvarTheta 825
\mitvartheta 865
\mitXi 821
\mitxi 848
\mitZeta 813
\mitzeta 840
\mode_if_math:F 996
\mode_if_math:TF 1708
\mskip 1468
\Mu 819
\mu 846
\muskip_gset:Nn 1463
\muskip_new:N 1462

N

\N 1056
\ne 1678
\neq 1678
\new@mathalphabet 330
\new@mathgroup 301, 365, 366
\new@mathversion 321
\new@symbolfont 325
\newcommand 646, 658, 673, 702, 807
\newcounter 7
\newfam 366
\newif 8–21
\newmathalphabet 306
\newmathalphabet@@ 307
\newmathalphabet@@@ 308
\newmcodes@ 1695
\noexpand 427, 665
\nolimits 383
\non@alpherr 998
\npolint 655
\Nu 820
\nu 847
\num_incr:N 1492
\num_new:N 1464
\num_zero:N 1488
\number 785, 792, 793
\numexpr 741, 746, 750, 751, 763, 765, 785, 792, 793

O

\o 1080
\oiiint 652
\oiint 652
\oint 652
\ointctrclockwise 653
\Omega 832
\omega 861
\Omicron 822
\omicron 849
\operator@font 1673
\or 147, 160, 173, 187, 193, 199, 215, 217, 241, 259, 277, 285, 292, 294

\overrightarrow 1675

P

\P 1057
\PackageError 678
\PackageInfo 456
\PackageWarningNoLine . . 448, 969, 1719
\peek_after:NN 1634
\peek_meaning_remove:NTF 1493, 1496, 1499
\Phi 829
\phi 856, 1613
\Pi 823
\pi 850
\pointint 655
\prg_case_int:nnn 1472
\prg_do_nothing: 984
\prg_new_conditional:Nnn 986
\prg_replicate:nn 1468
\prg_return_false: 987
\prg_return_true: 987
\prg_stepwise_variable:nnnNn 761, 790
\prime 1507
\primedouble 1475
\primequadruple 1481
\primesingle 535, 1467, 1468, 1473
\primetriple 1478
\process@table 320
\ProcessOptionsX 299
\prop_get:cxN 1664
\prop_get:NnN 1529
\prop_gput:Nnn 1528
\prop_gput:Nxn 1541, 1574
\prop_if_in:cxTF 1660
\prop_if_in:NnTF 1530
\prop_new:N 1526, 1527
\protect 681
\ProvidesPackage 1
\Psi 831
\psi 860

Q

\Q 1058

R

\R 1059, 1074, 1088
\r@@t 1515
\raise 1518
\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
\removenolimits 661
\RequirePackage 3–5
\restore@mathversion 317
\Rho 824
\rho 851, 1612
\rightarrow 1674
\rootbox 1522
\rppolint 654

S

\sb 1580
\scan_stop: . . . 540, 541, 987, 1543, 1576
\scantokens 1435, 1544, 1577
\scpolint 655
\scriptscriptstyle 358
\scriptstyle 355
\set@@mathdelimiter 343
\set@mathaccent 334
\set@mathchar 336
\set@mathdelimiter 342
\set@mathsymbol 337
\setbox 1516
\setkeys 426
\SetMathAlphabet 331
\SetMathAlphabet@ 332
\setmathfont 414, 681
\SetSymbolFont 326
\SetSymbolFont@ 327
\sf@size 431, 435
\Sigma 826
\sigma 853
\sp 1547
\space 969, 998
\sqint 655
\sqrt 670, 1514
\sqrtsign 1514, 1516
\std@equal 1691
\std@minus 1690, 1700
\stepcounter 462

\string 377, 380, 730, 731
\strip@pt 352
\sumint 653

T

\Tau 827
\tau 854
\tf@size 430, 431
\Theta 815
\theta 842
\theum@fam 463
\thinmuskip 1463
\tilde 1710
\tl_map_inline:nn 300
\tl_put_right:cn 647
\tl_put_right:NV 1668
\tl_remove_in:Nn 349, 1689
\tl_rescan:nn 1440
\tl_set:Nn 223–228, 230–235, 247–249, 251–253, 265–267, 269–271, 291, 293, 295, 422–425, 455, 978–982, 1546, 1548, 1579, 1581
\tl_set:Nx 204–207, 463, 1630, 1632
\tl_set_eq:NN 1631
\tl_use:c 972
\to 1674
\token_if_eq_catcode_p:NN . 1641, 1642
\token_if_eq_meaning_p:NN 1643

U

\um@addto@mathmap 634, 641, 646
\um@backslash 707, 730
\um@char@num@range 417, 638, 719, 720, 722
\um@char@range 416, 454, 581, 697, 700, 703
\um@firstchar 706, 731
\um@firstof 729–731
\um@font 356, 359, 444, 445, 974, 987, 1517, 1519, 1520, 1523
\um@fontdimen@percent 351, 356, 359, 1519, 1520
\um@mathsymbol 367, 486
\um@mversion 420, 421
\um@nolimits 381, 651, 659, 667
\um@parse@range 716, 732
\um@parse@term 489, 525, 702
\um@radicals 387, 670
\um@resolve@greek 806
\um@scaled@apply 354, 1517, 1523
\um@scanactivedef 374, 1431
\um@scancharlet 1431, 1451
\um@set@mathsymbol 368, 369
\um@usv@bbLatin 33, 1053
\um@usv@bblatin 34, 1049
\um@usv@bbnum 32, 1063
\um@usv@bfDigamma 95, 1176, 1273
\um@usv@bfdigamma 102, 1195, 1284
\um@usv@bffrakLatin 66, 1287
\um@usv@bffraklatin 67, 1290
\um@usv@bfGreek 60
\um@usv@bfgreek 61
\um@usv@bfitGreek 64, 204, 605, 608, 891, 1179, 1244
\um@usv@bfitgreek 65, 205, 612, 620, 892, 1198, 1250
\um@usv@bfith 112, 1165, 1171
\um@usv@bfitLatin 62, 206, 595, 597, 887, 1156, 1237, 1249
\um@usv@bfitlatin 63, 207, 600, 602, 888, 1164, 1240
\um@usv@bfitNabla 123, 252, 510, 518, 1183, 1246
\um@usv@bfitnum 53
\um@usv@bfitpartial 129, 270, 514, 520, 1206, 1251
\um@usv@bfitvarepsilon 114, 613, 621, 1207, 1223, 1252
\um@usv@bfitvarkappa 116, 615, 623, 1209, 1225, 1254
\um@usv@bfitvarphi 117, 616, 624, 1210, 1226, 1255
\um@usv@bfitvarpi 119, 618, 626, 1212, 1228, 1257
\um@usv@bfitvarrho 118, 617, 625, 1211, 1227, 1256
\um@usv@bfitvarTheta 113, 606, 609, 1181, 1190, 1245
\um@usv@bfitvartheta 115, 614, 622, 1208, 1224, 1253
\um@usv@bfLatin 55
\um@usv@bflatin 57
\um@usv@bfNabla 122, 248, 509, 518, 1182, 1272
\um@usv@bfnum 51, 1151, 1234, 1260

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

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

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

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

\um_sf_latin_up_or_it_usv 224, 231, 1109
\um_sub_or_super:n . . 1547, 1580, 1623
\um_symfont_tl 455, 463, 469, 486, 531, 539, 575, 578, 634, 641, 764
\UnicodeMathSymbol 457, 464, 1450
\unless 704
\updefault 470
\upDigamma 1713
\updigamma 1712
\upint 656
\Upsilon 828
\upsilon 855
\use:c 967, 992, 1000
\use_none:n 467

V

\varepsilon 862
\varkappa 866
\varointclockwise 653
\varphi 867
\varpi 871
\varrho 870
\varsigma 852
\varTheta 825
\vartheta 865
\vec 1675
\version@elt 314
\version@list 313

W

\widehat 1709
\widetilde 1710

X

\xdef 667, 700
\XeTeXdelcode 393, 398, 575, 578
\XeTeXdelimiter 392, 397
\XeTeXmathaccent 402
\XeTeXmathchardef 376, 538
\XeTeXmathcode 394, 399, 404, 412
\XeTeXmathcodenum 541, 1543, 1576
\XeTeXradical 389
\Xi 821
\xi 848
\XKV@rm 440

Z

\Z 1060, 1089
\z@ 1516, 1519, 1520, 1524
\Zeta 813
\zeta 840
\zf@family 470
\zf@fontspec 427
\zf@update@ff 686, 689