Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/09/11 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

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.

2 Specification

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

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

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

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

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

Finally, maths versions must also be provided for. While I guess version selection in 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. This syntax will also hook into the fontspec font feature processing:

`\setmathfont[Range=`⟨*unicode range*⟩`,`⟨*font features*⟩`]{`⟨*font name*⟩`}`

where ⟨*unicode range*⟩ is a comma-separated list of unicode slots and ranges such as `{27D0-27EB,27FF,295B-297F}`. Furthermore, preset names ranges could be used, such as `MiscMathSymbolsA`, with such ranges based on unicode chunks. The amount of optimisation required here to achieve acceptable performance has yet to be determined. Techniques such as saving out unicode subsets based on ⟨*unicode range*⟩ data to be `\input` in the next LaTeX run are a possibility, but at this stage, performance without such measures seems acceptable.

2.2 Script and scriptscript fonts/features

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

Other fonts will possibly use entirely separate fonts. Both of these options must be taken into account. I hope this will be mostly automatic from the users' points of view. The `+ssty` feature can be detected and applied automatically, and appropriate optical size information embedded in the fonts will ensure this latter case. Fine tuning should be possible automatically with fontspec options. We might have to wait until MnMath, for example, before we really know.

3 Maths input

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

: TODO : describe alphabet inputs

3.1 Miscellanea

3.1.1 Primes

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

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

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

4 Package options

4.1 Math 'style'

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

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

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

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

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

this; to follow from the example: `\mathup{g}`. Maths alphabets commands such as `\mathup` are detailed later.

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

The `math-style` options' effects are shown in brief in table 1. Figure 1 on page 8 shows every character under the effect of this package option.

4.2 Bold switching

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

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

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`	$(\mathbf{a},\mathbf{z},\mathbf{B},\mathbf{X})$	$(\boldsymbol{\upalpha},\boldsymbol{\upbeta},\mathbf{\Gamma},\mathbf{\Xi})$

Table 3: The various forms of nabla.

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

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

The `bold-style` options' effects are shown in brief in table 2. Figure 2 on page 8 shows every character under the effect of this package option.

4.3 Symbols requiring special attention

Nabla The symbol ∇ comes in the six forms shown in table 3. We want an individual option to specify whether we want upright or italic nabla by default (when either upright or italic nabla is used in the source). TeX classically uses an upright nabla, but ISO standards differ (I think). The package options `nabla=upright` and `nabla=italic` switch between the two choices. This is then inherited through `\mathbf`; `\mathit` and `\mathup` can be used to force one way or the other.

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

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

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

Table 4: The various forms of the partial differential. Note that in the fonts used to display these glyphs, the first upright partial is incorrectly shown in an italic style.

Description		Glyph
Regular	Upright	∂
	Italic	𝜕
Bold	Upright	𝛛
	Italic	𝝏
Sans bold	Upright	⍰
	Italic	⍰

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

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

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

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

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

U+3B5: GREEK SMALL LETTER EPSILON
U+3F5: GREEK LUNATE EPSILON SYMBOL
U+3C6: GREEK SMALL LETTER PHI
U+3D5: GREEK SMALL LETTER SCRIPT PHI

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:

[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.

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘϴΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
αβγδεϵζηθϑικϰλμνξοπϖρϱςστυφϕχψω

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘ□ΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
αβγδεϵζηθϑικϰλμνξοπϖρϱςστυφϕχψω

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘ□ΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
αβγδϵ□ζηθϑικϰλμνξοπϖρϱςστυϕφχψω

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

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡϴΣΤΥΦΧΨΩ
αβγδεζηθικλμνξοπρςστυφχψωϵϑϰϕϱϖ

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡϴΣΤΥΦΧΨΩ
αβγδεζηθικλμνξοπρςστυφχψωϵϑϰϕϱϖ

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡϴΣΤΥΦΧΨΩ
αβγδεζηθικλμνξοπρςστυφχψωϵϑϰϕϱϖ

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

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

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

File I
The unicode-math package

This is the package.

```
\ProvidesPackage{unicode-math}
  [2009/09/11 v0.4 Unicode maths in XeLaTeX]
```

5 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
```

5.0.1 Alphabet unicode positions

Before we begin, let's define the positions of the various unicode alphabets so that our code is a little more readable.[2]

```
\def\um@usv@num{`\0}
\def\um@usv@upLatin{`\A}
\def\um@usv@uplatin{`\a}
\def\um@usv@itLatin{"1D434}
\def\um@usv@itlatin{"1D44E}
\def\um@usv@upGreek{"391}
\def\um@usv@upgreek{"3B1}
\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@sfLatin{"1D5A0}
\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@bfLatin{"1D400}
\def\um@usv@bflatin{"1D41A}
\let\um@usv@bfuplatin\um@usv@bflatin
\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}
```

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

```
\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@bfsfLatin{"1D5D4}
\def\um@usv@bfsflatin{"1D5EE}
\let\um@usv@bfsfuplatin\um@usv@bfsflatin
\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:

```
\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}
```

5.1 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_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_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_false:N \g_um_texgreek_bool
  \or
    \@um@literaltrue
    \@um@bfliteraltrue
    \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@uppartialfalse
    \@um@bfupLatinfalse
    \@um@bfuplatinfalse
  \or
    \@um@bfupGreektrue
    \@um@bfupgreekfalse
    \@um@uppartialfalse
    \@um@bfupLatintrue
    \@um@bfuplatintrue
  \or
    \@um@bfupGreektrue
    \@um@bfupgreektrue
    \@um@uppartialtrue
    \@um@bfupLatintrue
    \@um@bfuplatintrue
  \or
    \@um@bfliteraltrue
  \fi
}
```

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
}
```

```
\ExecuteOptionsX{math-style=TeX}
\ProcessOptionsX
```

5.2 Overcoming `\@onlypreamble`

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

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

5.3 Other things

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

`\fontdimen`s `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
}
```

6 Fundamentals

6.1 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`.

6.2 `\DeclareMathSymbol` for unicode ranges

This command is a bit funny at the moment; it doesn't define the actual macro for almost all of the symbols passed to it, but it does assign the `\XeTeXmathchar`.

`\um@mathsymbol`

`#1` : Symbol, *e.g.*, `\alpha`
`#2` : Type, *e.g.*, `\mathalpha`
`#3` : Math font name, *e.g.*, `operators`
`#4` : Slot, *e.g.*, `"221E`

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

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

```
    \begingroup
      \char_make_active:n {#4}
      \global\mathcode#4="8000\relax
      \um@scanactivedef #4 \@nil { \csname\string#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 \sum@op as \sum@sym, followed by \nolimits if necessary.

```
    \cs_gset:cpn { \string#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
}
```

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

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

A

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

6.3 The main \setmathfont macro

Here's the simplest usage:

$Ax \overset{\text{def}}{=} \nabla \times \mathscr{Z}$

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

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

$$F(s) = \mathscr{L}\{f(t)\} = \int_0^\infty \mathrm{e}^{-st} f(t)\,\mathrm{d}t$$

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

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 `\fontdimen`s:

```
\font\um@font="#2"\relax
\ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
  \@um@ot@math@true
\else
  \PackageWarning{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 §6.3.1 for the individual definitions

```
\ifx\um@char@range\@empty
  \def\um@symfont{um@allsym}
 \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
  \let \UnicodeMathSymbol \um@mathsymbol@noparse
  \let \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}
    \edef\um@symfont{um@fam\theum@fam}
    \let \UnicodeMathSymbol \um@mathsymbol@parse
    \let \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` as the LaTeX math font to access everything:

```
  \DeclareSymbolFont{\um@symfont}
    {\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_alphanum:
  \um_setup_alphabets:
```

End of the `\setmathfont` macro.

```
}
```

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

6.3.1 Functions for setting up symbols with mathcodes

`\um@mathsymbol@noparse`

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

\um@mathsymbol@parse

If the Range font feature has been used, then only a subset of the unicode glyphs are to be defined. See section §7.3 for the code that enables this.

```
\newcommand\um@mathsymbol@parse[4]{
  \um@parse@term{#1}{#2}{#3}{
    \um@mathsymbol{#2}{#3}{\um@symfont}{#1}
  }
}
```

\um_remap_symbols:

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{"2D}{\mathbin}{"02212}% hyphen to minus
  \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} {
    \SetMathCode {##1} {#2} {\um@symfont} {#3}
  }
}
```

6.3.2 Active math characters

\um_setup_mathactives:

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

\um_make_mathactive:nNN Makes #1 a mathactive char, and gives cs #2 the meaning of mathchar #1 with class #3. You are responsible for giving active #1 a particular meaning!

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

6.3.3 Maths alphabets’ character mapping

We want it to be convenient for users to actually type in maths. The ascii Latin characters should be used for italic maths, and the text Greek characters should be used for upright/italic (depending on preference) Greek, if desired.

\um_setup_alphanum: All symbols input that aren’t defined directly in `unicode-math-table`.

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

Normal weight

```
    \if@um@literal
      \um_setup_literals:
    \else
      \if@um@upLatin\um@def@upLatin\else\um@def@itLatin\fi
      \if@um@uplatin\um@def@uplatin\else\um@def@itlatin\fi
      \if@um@upGreek\um@def@upGreek\else\um@def@itGreek\fi
      \if@um@upgreek\um@def@upgreek\else\um@def@itgreek\fi
    \fi
```

Bold

```
  \if@um@bfliteral
   \um_setup_bf_literals:
  \else
   \if@um@bfupLatin
   \um@setmathcode[26]{\um@usv@bfLatin,\um@usv@bfitLatin}{\um@usv@bfLatin}
   \else
   \um@setmathcode[26]{\um@usv@bfLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
   \fi
   \if@um@bfuplatin
   \um@setmathcode[26]{\um@usv@bflatin,\um@usv@bfitlatin}{\um@usv@bflatin}
   \else
   \um@setmathcode[26]{\um@usv@bflatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
   \fi
   \if@um@bfupGreek
   \um@setmathcode[25]{\um@usv@bfGreek,\um@usv@bfitGreek}{\um@usv@bfGreek}
   \um@setmathcode{\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
   \else
   \um@setmathcode[25]{\um@usv@bfGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}
   \um@setmathcode{\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
   \fi
   \if@um@bfupgreek
   \um@setmathcode[25]{\um@usv@bfgreek,\um@usv@bfitgreek}{\um@usv@bfgreek}
   \um@setmathcode{\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
   \um@setmathcode{\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
   \um@setmathcode{\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
   \um@setmathcode{\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
   \um@setmathcode{\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
   \um@setmathcode{\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
   \else
   \um@setmathcode[25]{\um@usv@bfgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
   \um@setmathcode{\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
   \um@setmathcode{\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
   \um@setmathcode{\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
   \um@setmathcode{\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
   \um@setmathcode{\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
   \um@setmathcode{\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
   \fi
  \fi
 \else
```

: TODO : what is supposed to happen here?

```
 \fi
}
```

6.3.4 Functions for setting up the maths alphabets

\um_mathmap_noparse:Nnn

#1 : Maths alphabet, *e.g.*, \mathbb
#2 : Input slot(s), *e.g.*, the slot for 'A' (comma separated)
#3 : Output slot, *e.g.*, the slot for '𝔸'

Adds \SetMathCode declaractions 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 {##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 \SetMathCode 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 {####1}{#1}{#3}
      }
    \fi
  }
}
```

\um@addto@mathmap

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

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

```
\newcommand\um@addto@mathmap[4]{
  \expandafter\g@addto@macro
    \csname um_setup_\cs_to_str:N #3:\endcsname{
    \SetMathCode{#2}{\mathalpha}{#1}{#4}
  }
}
```

6.4 (Big) operators

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

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

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

USV	Ex.	Macro	Description
U+02140	$\Bbbsum\limits_0^1$	`\Bbbsum`	DOUBLE-STRUCK N-ARY SUMMATION
U+0220F	$\prod\limits_0^1$	`\prod`	PRODUCT OPERATOR
U+02210	$\coprod\limits_0^1$	`\coprod`	COPRODUCT OPERATOR
U+02211	$\sum\limits_0^1$	`\sum`	SUMMATION OPERATOR
U+0222B	$\int_0^1$	`\int`	INTEGRAL OPERATOR
U+0222C	$\iint_0^1$	`\iint`	DOUBLE INTEGRAL OPERATOR
U+0222D	$\iiint_0^1$	`\iiint`	TRIPLE INTEGRAL OPERATOR
U+0222E	$\oint_0^1$	`\oint`	CONTOUR INTEGRAL OPERATOR
U+0222F	$\oiint_0^1$	`\oiint`	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230	$\oiiint_0^1$	`\oiiint`	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231	$\intclockwise_0^1$	`\intclockwise`	CLOCKWISE INTEGRAL
U+02232	$\varointclockwise_0^1$	`\varointclockwise`	CONTOUR INTEGRAL, CLOCKWISE
U+02233	$\ointctrclockwise_0^1$	`\ointctrclockwise`	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0	$\bigwedge\limits_0^1$	`\bigwedge`	LOGICAL OR OPERATOR
U+022C1	$\bigvee\limits_0^1$	`\bigvee`	LOGICAL AND OPERATOR
U+022C2	$\bigcap\limits_0^1$	`\bigcap`	INTERSECTION OPERATOR
U+022C3	$\bigcup\limits_0^1$	`\bigcup`	UNION OPERATOR
U+027D5	$\leftouterjoin\limits_0^1$	`\leftouterjoin`	LEFT OUTER JOIN
U+027D6	$\rightouterjoin\limits_0^1$	`\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 16). 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
}
```

$\iiint_V$ $\iiint\limits_V$

$\iiint_V$

```
\def\dmath#1{$\displaystyle #1$}
\setmathfont{Asana Math} \dmath{\iiint_V}
\removenolimits\iiint
\setmathfont{Asana Math} \dmath{\iiint_V}
\addnolimits\iiint
\setmathfont{Asana Math} \dmath{\iiint_V}
```

6.5 Radicals

The radical for square root is organised in `\um@set@mathsymbol` on page **??**. I think it's the only radical ever. (Actually, there is also `\cuberoot` and `\fourthroot`, but they don't seem to behave as proper radicals.)

Also, what about right-to-left square roots?

`\um@radicals` We organise radicals in the same way as nolimits-operators; that is, in a comma-list.

```
\def\um@radicals{\sqrt}
```

$$\sqrt{1+\sqrt{1+\sqrt{1+\sqrt{1+\sqrt{1+\sqrt{1+\sqrt{1+x}}}}}}}$$

```
\[ \sqrt{1+\sqrt{1+
 \sqrt{1+ \sqrt{1+
 \sqrt{1+\sqrt{1+
 \sqrt{1+x}}}}}}} \]
```

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

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

6.6 Delimiters

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

`http://www.latex-project.org/cgi-bin/ltxbugs2html?pr=latex/3853&prlatex/3754`

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

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

: TODO : 'fences', e.g., `\vert`

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

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

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

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

Here are all `\mathopen` characters:

USV	Ex.	Macro	Description
U+00028	(	`\lparen`	LEFT PARENTHESIS
U+0005B	[	`\lbrack`	LEFT SQUARE BRACKET
U+0007B	{	`\lbrace`	LEFT CURLY BRACKET
U+000AB	«	`\guillemotleft`	DOUBLE ANGLE QUOTATION MARK (GUILLEMET), LEFT
U+02018	‘	`\lq`	SINGLE QUOTATION MARK, LEFT
U+0201A	‚	`\quotsinglbase`	RISING SINGLE QUOTE, LEFT (LOW)
U+0201E	„	`\quotdblbase`	RISING DOUBLE QUOTE, LEFT (LOW)
U+02039	‹	`\guilsinglleft`	SINGLE ANGLE QUOTATION MARK (GUILLEMET), LEFT
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
U+027E6	⟦	\lBrack	MATHEMATICAL LEFT WHITE SQUARE BRACKET
U+027E8	⟨	\langle	MATHEMATICAL LEFT ANGLE BRACKET
U+027EA	⟪	\lAngle	MATHEMATICAL LEFT DOUBLE ANGLE BRACKET
U+027EC		\Lbrbrak	MATHEMATICAL LEFT WHITE TORTOISE 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
U+0298D	⦍	\lbrackultick	LEFT SQUARE BRACKET WITH TICK IN TOP CORNER
U+0298F	⦏	\lbracklltick	LEFT SQUARE BRACKET WITH TICK IN 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+0007D	}	\rbrace	RIGHT CURLY BRACKET
U+000BB	»	\guillemotright	DOUBLE ANGLE QUOTATION MARK (GUILLEMET), RIGHT
U+02019	’	\rq	SINGLE QUOTATION MARK, RIGHT
U+0201B	‛	\quotsinglright	RISING SINGLE QUOTE, RIGHT (HIGH)
U+0201F	‟	\quotdblright	RISING DOUBLE QUOTE, RIGHT (HIGH)
U+0203A	›	\guilsinglright	SINGLE ANGLE QUOTATION MARK (GUILLEMET), RIGHT
U+02309	⌉	\rceil	RIGHT CEILING
U+0230B	⌋	\rfloor	RIGHT FLOOR
U+0231D	⌝	\urcorner	UPPER RIGHT CORNER
U+0231F	⌟	\lrcorner	LOWER RIGHT CORNER
U+02773		\rbrbrak	LIGHT RIGHT TORTOISE SHELL BRACKET ORNAMENT

U+027C6	⟆	`\rbag`	RIGHT S-SHAPED BAG DELIMITER
U+027E7	⟧	`\rBrack`	MATHEMATICAL RIGHT WHITE SQUARE BRACKET
U+027E9	⟩	`\rangle`	MATHEMATICAL RIGHT ANGLE BRACKET
U+027EB	⟫	`\rAngle`	MATHEMATICAL RIGHT DOUBLE ANGLE BRACKET
U+027ED		`\Rbrbrak`	MATHEMATICAL RIGHT WHITE TORTOISE SHELL BRACKET
U+02984	⦄	`\rBrace`	RIGHT WHITE CURLY BRACKET
U+02986	⦆	`\rParen`	RIGHT WHITE PARENTHESIS
U+02988	⦈	`\rrparenthesis`	Z NOTATION RIGHT IMAGE BRACKET
U+0298A	⦊	`\rrangle`	Z NOTATION RIGHT BINDING BRACKET
U+0298C	⦌	`\rbrackubar`	RIGHT SQUARE BRACKET WITH UNDERBAR
U+0298E	⦎	`\rbracklrtick`	RIGHT SQUARE BRACKET WITH TICK IN BOTTOM CORNER
U+02990	⦐	`\rbrackurtick`	RIGHT SQUARE BRACKET WITH TICK IN TOP 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

6.7 Maths accents

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

USV	Ex.	Macro	Description
U+00300	□	`\grave`	GRAVE ACCENT
U+00301	□	`\acute`	ACUTE ACCENT
U+00302	□	`\hat`	CIRCUMFLEX ACCENT
U+00303	□	`\tilde`	TILDE
U+00304	□	`\bar`	MACRON
U+00305	□	`\overbar`	OVERBAR EMBELLISHMENT
U+00306	□	`\breve`	BREVE
U+00307	□	`\dot`	DOT ABOVE
U+00308	□	`\ddot`	DIERESIS
U+00309	□	`\ovhook`	COMBINING HOOK ABOVE
U+0030A	□	`\ocirc`	RING
U+0030C	□	`\check`	CARON
U+00310	□	`\candra`	CANDRABINDU (NON-SPACING)

U+00312	☒	\oturnedcomma	COMBINING TURNED COMMA ABOVE
U+00313	☒	\osmooth	GREEK PSILI (SMOOTH BREATHING) (NON-SPACING)
U+00314	☒	\orough	GREEK DASIA (ROUGH BREATHING) (NON-SPACING)
U+00315	☒	\ocommatopright	COMBINING COMMA ABOVE RIGHT
U+0031A	☒	\droang	LEFT ANGLE ABOVE (NON-SPACING)
U+020D0	x⃐	\leftharpoonaccent	COMBINING LEFT HARPOON ABOVE
U+020D1	x⃑	\rightharpoonaccent	COMBINING RIGHT HARPOON ABOVE
U+020D2	x⃒	\vertoverlay	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6	x⃖	\overleftarrow	COMBINING LEFT ARROW ABOVE
U+020D7	x⃗	\vec	COMBINING RIGHT ARROW ABOVE
U+020DB	x⃛	\dddot	COMBINING THREE DOTS ABOVE
U+020DC	x⃜	\ddddot	COMBINING FOUR DOTS ABOVE
U+020E1	x⃡	\overleftrightarrow	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7	☒	\annuity	COMBINING ANNUITY SYMBOL
U+020E8	x⃨	\threeunderdot	COMBINING TRIPLE UNDERDOT
U+020E9	x⃩	\widebridgeabove	COMBINING WIDE BRIDGE ABOVE
U+020EC	x⃬	\underrightharpoondown	COMBINING RIGHTWARDS HARPOON WITH BARB DOWNWARDS
U+020ED	x⃭	\underleftharpoondown	COMBINING LEFTWARDS HARPOON WITH BARB DOWNWARDS
U+020EE	x⃮	\underleftarrow	COMBINING LEFT ARROW BELOW
U+020EF	x⃯	\underrightarrow	COMBINING RIGHT ARROW BELOW
U+020F0	☒	\asteraccent	COMBINING ASTERISK ABOVE

7 Font features

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

```
\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
  }
}
```

7.1 OpenType maths font features

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

7.2 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}{}
```

7.3 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 9.

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

Input	Range
`x`	$r = x$
`x-`	$r \geqslant x$
`-y`	$r \leqslant y$
`x-y`	$x \leqslant r \leqslant y$

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 9, 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@setmathcode

#1 : Starting input char(s)
#2 : Number of iterations
#3 : Starting output char

Loops through character ranges setting \mathcode.

```
\newcommand\um@setmathcode[3][1]{
  \clist_map_variable:nNn {#2} \l_um_input_num {
    \prg_stepwise_variable:nnnNn{1}{1}{#1} \l_um_incr_num {
      \SetMathCode
        {\numexpr \l_um_incr_num+ \l_um_input_num - 1\relax}
        {\mathalpha}{\um@symfont}
        {\numexpr \l_um_incr_num + #3 - 1\relax}
    }
  }
}
```

\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}
}
```

BCDBCD ABCDEF

```
{\um@setmathcode[3]{`\A,`\D}{`\B}
$ABCDEF$} $ABCDEF$
```

\um@resolve@greek This macro defines \Alpha...\omega as their corresponding unicode (mathematical italic) character. Remember that the mapping to upright or italic happens with the mathcode definitions, whereas these macros just stand for the literal unicode characters.

```
\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}
}
```

`\um@def@numbers`

```
\newcommand\um@def@numbers{
  \um@setmathcode[10]{\um@usv@num}{\um@usv@num}
}
```

`\um_setup_literals:` : TODO : other literal symbols

```
\cs_set:Nn \um_setup_literals: {
  \um@setmathcode[26]{\um@usv@upLatin}{\um@usv@upLatin}
  \um@setmathcode[26]{\um@usv@itLatin}{\um@usv@itLatin}
  \um@setmathcode[26]{\um@usv@itlatin}{\um@usv@itlatin}
  \um@setmathcode{\um@usv@ith}{\um@usv@ith}
  \um@setmathcode[26]{\um@usv@uplatin}{\um@usv@uplatin}
  \um@setmathcode[25]{\um@usv@upGreek}{\um@usv@upGreek}
  \um@setmathcode{\um@usv@varTheta}{\um@usv@varTheta}
  \um@setmathcode[25]{\um@usv@itGreek}{\um@usv@itGreek}
  \um@setmathcode[25]{\um@usv@upgreek}{\um@usv@upgreek}
}
```

`\um_setup_bf_literals:` TODO: other literal symbols

```
\cs_set:Nn \um_setup_bf_literals: {
  \um@setmathcode[26]{\um@usv@bfLatin}{\um@usv@bfLatin}
  \um@setmathcode[26]{\um@usv@bflatin}{\um@usv@bflatin}
  \um@setmathcode[26]{\um@usv@bfitLatin}{\um@usv@bfitLatin}
  \um@setmathcode[26]{\um@usv@bfitlatin}{\um@usv@bfitlatin}
  \um@setmathcode[25]{\um@usv@bfGreek}{\um@usv@bfGreek}
  \um@setmathcode[25]{\um@usv@bfgreek}{\um@usv@bfgreek}
  \um@setmathcode[25]{\um@usv@bfitGreek}{\um@usv@bfitGreek}
  \um@setmathcode[25]{\um@usv@bfitgreek}{\um@usv@bfitgreek}
}
```

`\um@def@upLatin`

```
\newcommand\um@def@upLatin{
  \um@setmathcode[26]{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
}
```

`\um@def@itLatin`

```
\newcommand\um@def@itLatin{
  \um@setmathcode[26]{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
}
```

`\um@def@itlatin` Don't overlook 'h', which maps to u+210e: planck constant instead of the expected u+1d455: mathematical italic small h.

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

\um@def@uplatin

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

\um@def@upGreek

```
\newcommand\um@def@upGreek{
  \um@setmathcode[25]{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
  \um@setmathcode{\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
}
```

\um@def@itGreek

```
\newcommand\um@def@itGreek{
  \um@setmathcode[25]{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
  \um@setmathcode{\um@usv@varTheta}{\um@usv@itvarTheta}
}
```

\um@def@upgreek

```
\newcommand\um@def@upgreek{
  \um@setmathcode[25]{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
 \um@setmathcode{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
  \um@setmathcode{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
  \um@setmathcode{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
  \um@setmathcode{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
  \um@setmathcode{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
  \um@setmathcode{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
}
```

\um@def@itgreek

```
\newcommand\um@def@itgreek{
  \um@setmathcode[25]{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
 \um@setmathcode{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
 \um@setmathcode{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
 \um@setmathcode{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
  \um@setmathcode{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
  \um@setmathcode{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
  \um@setmathcode{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
}
```

8 Maths alphabets mapping definitions

Algorithm for setting alphabet fonts:

- By default, try and set all of them.

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

```
\cs_new:Nn \um_setup_math_alphabet:n {
  \um_glyph_if_exist:nTF {\csname um@usv@#1latin \endcsname}{
    \um_maybe_init_alphabet:n {#1}
    \um_prepare_alph:n {#1}
    \use:c {um_config_math#1:}
  }{
   \PackageWarning{unicode-math}{Math~ alphabet~ "#1"~ not~ found~ with~ this~ font}
  }
}
```

```
\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.

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

```
\cs_new:Nn \um_setup_alphabets: {
  \um_setup_math_alphabet:n {up     }
  \um_setup_math_alphabet:n {it     }
  \um_setup_math_alphabet:n {bb    }
  \um_setup_math_alphabet:n {scr   }
  \um_setup_math_alphabet:n {frak  }
  \um_setup_math_alphabet:n {sf    }
  \um_setup_math_alphabet:n {sfit  }
  \um_setup_math_alphabet:n {tt    }
  \um_setup_math_alphabet:n {bf    }
  \um_setup_math_alphabet:n {bfup  }
  \um_setup_math_alphabet:n {bfit  }
  \um_setup_math_alphabet:n {bfscr }
  \um_setup_math_alphabet:n {bffrak}
  \um_setup_math_alphabet:n {bfsf  }
  \um_setup_math_alphabet:n {bfsfup}
  \um_setup_math_alphabet:n {bfsfit}
}
```

: TODO : nested alphabets?

8.0.1 Upright: \mathup

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
□
αβγδϵζηθικλμνξοπρστυφχψω
□ϑϰφϱϖ

```
$\mathup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathup{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathup{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad$\mathup{□}$ \\
$\mathup{αβγδεζηθικλμνξοπρστυφχψω}$\quad$\mathup{□□□□□□}$ \\
```

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

```
\cs_new:Npn \um_config_mathup: {
 \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
 \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
 \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
 \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@partial,\um@usv@itpartial}{\um@usv@partial}
 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varThe
 \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}
}
```

8.0.2 Italic: `\mathit`

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
ϴ
αβγδεζηθικλμνξοπρστυφχψω
ϵϑϰϕϱϖ

```
$\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathit{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathit{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad$\mathit{□}$ \\
$\mathit{αβγδεζηθικλμνξοπρστυφχψω}$\quad$\mathit{□□□□□□}$ \\
```

Roman:

```
\cs_new:Npn \um_config_mathit: {
  \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
  \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}
```

Greek:

```
  \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
  \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartia
  \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarT
  \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}
}
```

8.0.3 Blackboard or double-struck: `\mathbb`

$\mathbb{0123456789}$
$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
$\mathbb{abcdefghijklmnopqrstuvwxyz}$

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

Numbers:

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

Roman uppercase:

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

Roman lowercase:

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

8.0.4 Script or caligraphic: \mathscr and \mathcal

𝒜ℬ𝒞𝒟ℰℱ𝒢ℋℐ𝒥𝒦ℒℳ𝒩𝒪𝒫𝒬ℛ𝒮𝒯𝒰𝒱𝒲𝒳𝒴𝒵
𝒶𝒷𝒸𝒹ℯ𝒻ℊ𝒽𝒾𝒿𝓀𝓁𝓂𝓃ℴ𝓅𝓆𝓇𝓈𝓉𝓊𝓋𝓌𝓍𝓎𝓏

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

```
\cs_new:Npn \um_config_mathscr: {
 \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}
 \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}
}
```

8.0.5 Fractur or fraktur or blackletter: \mathfrak

𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔ℜ𝔖𝔗𝔘𝔙𝔚𝔛𝔜□
𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷

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

Letters, with exceptions {ℭ, ℌ, ℑ, ℜ, □}:

```
\cs_new:Npn \um_config_mathfrak: {
 \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}
  \um_set_mathalphabet_latin:Nnn{\mathfrak}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklat
}
```

8.0.6 Sans serif: \mathsf

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

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

```
\cs_new:Npn \um_config_mathsf: {
  \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnum}
  \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfLatin}
  \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sflatin}
}
```

8.0.7 Sans serif italic: \mathsfit

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

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

```
\cs_new:Npn \um_config_mathsfit: {
  \um_set_mathalphabet_numbers:Nnn{\mathsfit}{\um@usv@num}{\um@usv@sfnum}
  \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLat
  \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlat
}
```

8.0.8 Typewriter or monospaced: \mathtt

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

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

```
\cs_new:Npn \um_config_mathtt: {
  \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
  \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
  \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlatin}
}
```

8.1 Bold alphabets' character mappings

8.1.1 Bold: \mathbf

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
ϴ□
αβγδεζηθικλμνξοπρστυφχψω
ϵϑϰϕϱϖ□

```
$\mathbf{0123456789}$ \\
$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbf{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbf{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad$\mathbf{□□}$ \\
$\mathbf{αβγδεζηθικλμνξοπρστυφχψω}$\quad$\mathbf{□□□□□□□}$ \\
```

```
\cs_new:Npn \um_config_mathbf: {
  \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{"1D7CA}
  \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{"1D7CB}
  \if@um@bfliteral
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfLatin}
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@bflatin}
   \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@bfitlatin}
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfGreek}
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfgreek}
   \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitgreek}
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
   \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@digamma}{\um@usv@bfdigamma}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
   \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
    \if@um@bfupLatin
    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfLatin
    \else
    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLat
    \fi
    \if@um@bfuplatin
    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bflatin
      \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
    \else
    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlat
      \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
    \fi
    \if@um@bfupGreek
    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfGreek
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfva
    \else
    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGre
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfit
    \fi
    \if@um@bfupgreek
    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfgreek
    \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}
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpart
    \else
    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgre
    \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}
    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpa
    \fi
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um_bfNabla_up_or_i
   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um_bfpartial_u
  \fi
}
```

8.1.2 Bold Italic: \mathbfit

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ ϴ
αβγδεζηθικλμνξοπρστυφχψω ϵϑϰϕϱϖ

```
$\mathbfit{0123456789}$ \\
$\mathbfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfit{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbfit{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad
  $\mathbfit{⍰}$ \\
$\mathbfit{αβγδεζηθικλμνξοπρστυφχψω}$\quad
  $\mathbfit{⍰⍰⍰⍰⍰⍰}$ \\
```

```
\cs_new:Npn \um_config_mathbfit: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLat
  \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_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgre
  \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfLatin}{\um@usv@bfitLatin}
  \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bflatin}{\um@usv@bfitlatin}
  \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfGreek}{\um@usv@bfitGreek}
  \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfgreek}{\um@usv@bfitgreek}
  \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}
  \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}
}
```

8.1.3 Bold Italic: \mathbfup

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ ϴ
αβγδεζηθικλμνξοπρστυφχψω ϵϑϰϕϱϖ

```
$\mathbfup{0123456789}$ \\
$\mathbfup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfup{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbfup{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad
  $\mathbfup{⍰}$ \\
$\mathbfup{αβγδεζηθικλμνξοπρστυφχψω}$\quad
  $\mathbfup{⍰⍰⍰⍰⍰⍰}$ \\
```

```
\cs_new:Npn \um_config_mathbfup: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfLatin
  \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bflatin
  \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfGreek
  \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfgreek
  \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bfLatin}{\um@usv@bfLatin}
  \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bflatin}{\um@usv@bflatin}
  \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfGreek}{\um@usv@bfGreek}
  \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfgreek}{\um@usv@bfgreek}
  \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@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}
}
```

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

𝕬𝕭𝕮𝕯𝕰𝕱𝕲𝕳𝕴𝕵𝕶𝕷𝕸𝕹𝕺𝕻𝕼𝕽𝕾𝕿𝖀𝖁𝖂𝖃𝖄𝖅
𝖆𝖇𝖈𝖉𝖊𝖋𝖌𝖍𝖎𝖏𝖐𝖑𝖒𝖓𝖔𝖕𝖖𝖗𝖘𝖙𝖚𝖛𝖜𝖝𝖞𝖟

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

```
\cs_new:Npn \um_config_mathbffrak: {
  \um_set_mathalphabet_numbers:Nnn{\mathbffrak}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin, \um@usv@itLatin,\um@usv@frakLa
  \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin,\um@usv@frakla
}
```

8.1.5 Bold script or calligraphic: `\mathbfscr`

𝓐𝓑𝓒𝓓𝓔𝓕𝓖𝓗𝓘 𝓙𝓚𝓛𝓜𝓝𝓞𝓟𝓠𝓡𝓢𝓣𝓤𝓥𝓦𝓧𝓨𝓩
𝓪𝓫𝓬𝓭𝓮𝓯𝓰𝓱𝓲𝓳𝓴𝓵𝓶𝓷𝓸𝓹𝓺𝓻𝓼𝓽𝓾𝓿𝔀𝔁𝔂𝔃

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

```
\cs_new:Npn \um_config_mathbfscr: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfscr}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrL
  \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrl
}
```

8.1.6 Bold sans serif: \mathbfsf

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ ϴ
αβγδεζηθικλμνξοπρστυφχψω εϑϰϕϱϖ

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

: TODO : These should be contextual!

Numbers (always upright) and letters:

```
\cs_new:Npn \um_config_mathbfsf: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfsf}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_latin:Nnn{\mathbfsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfLat
  \um_set_mathalphabet_latin:Nnn{\mathbfsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsflat
  \um_set_mathalphabet_greek:Nnn{\mathbfsf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfGre
  \um_set_mathalphabet_greek:Nnn{\mathbfsf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfgre
```

Others:

```
  \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varTheta,\um@usv@itvarTheta}{"1D767}
  \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@Nabla,\um@usv@itNabla}{"1D76F}
  \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@partial,\um@usv@itpartial}{"1D789}
  \um_set_mathalphabet_char:Nnn{\mathbfsf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D78A}
  \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}
}
```

8.1.7 Bold upright sans serif: \mathbfsfup

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ ϴ
αβγδεζηθικλμνξοπρστυφχψω εϑϰϕϱϖ

```
\setmathfont{STIXGeneral-Bold}
$\mathbfsfup{0123456789}$ \\
$\mathbfsfup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfsfup{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbfsfup{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad
  $\mathbfsfup{□}$ \\
$\mathbfsfup{αβγδεζηθικλμνξοπρστυφχψω}$\quad
  $\mathbfsfup{□□□□□□}$ \\
```

Numbers (always upright) and letters:

```
\cs_new:Npn \um_config_mathbfsfup: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfsfup}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfL
  \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfl
  \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfG
  \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfg
```

Others:

```
  \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}
  \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}
}
```

8.1.8 Bold italic sans serif: `\mathbfsfit`

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ   ϴ
αβγδεζηθικλμνξοπρστυφχψω   ϵϑϰϕϱϖ

```
\setmathfont{STIXGeneral-BoldItalic}
$\mathbfsfit{0123456789}$ \\
$\mathbfsfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfsfit{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbfsfit{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$\quad
  $\mathbfsfit{□}$ \\
$\mathbfsfit{αβγδεζηθικλμνξοπρστυφχψω}$\quad
  $\mathbfsfit{□□□□□□}$ \\
```

```
\cs_new:Npn \um_config_mathbfsfit: {
  \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@num}{\um@usv@bfnum}
  \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfi
  \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfi
  \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfi
  \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfi
```

Other symbols:

```
   \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
  \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}
}
```

8.2 Definitions of the math symbols

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

`\um@scancharlet`
`\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 \^
  \gdef\um@scancharlet#1="#2\@nil{
    \lowercase{
      \scantokens{\global\let#1=^^^^^#2}
    }
  }
  \gdef\um@scanactivedef"#1\@nil#2{
    \lowercase{
      \scantokens{\global\def^^^^^#1{#2}}
    }
  }
\endgroup
\let\unicodemathgobble\@gobble
```

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

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

8.3 Epilogue

Lots of little things to tidy up.

8.3.1 Unicode radicals

Undo the damage made to `\sqrt`:

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

8.3.2 Primes

$$[x']\ [x''']\ [x'''''']$$
$$[x']\ [x''']\ [x'''''']$$
$$[x']\ [x''']\ [x'''''']$$

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

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

U+2032: PRIME (`\primesingle`): x'

U+2033: DOUBLE PRIME (`\primedouble`): x''

U+2034: TRIPLE PRIME (`\primetriple`): x'''

U+2057: QUADRUPLE PRIME (`\primequadruple`): x''''

As you can see, they're all drawn at the correct height without being superscripted. However, in a correctly behaviour OpenType font with the MATH table, we also see different behaviour after the `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\prime` or `x^\prime` 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 `\prime`s 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.

Insert a \bgroup…\egroup wrapper so that superscript primes work, but does this break spacing for the rest of the time?

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

8.3.3 Radicals

`\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@
}
```

8.3.4 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\le{\leq}
\def\ge{\geq}
```

`\mathcal`

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

`\mathrm`

```
\def\mathrm{\mathup}
```

Overriding amsmath definitions:

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

Interaction with beamer:

```
\AtBeginDocument{
  \@ifpackageloaded{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 XƎTEX, 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 $ABC - xyz$, $\mathfrak{ABC} - \mathcal{XYZ}$, 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.

\fontdimen	Dimension name	Description
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.
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	SuperscriptBaselineDropMax	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	SuperscriptBottomMaxWithSubscript	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.

\fontdimen	Dimension name	Description
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.
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.

\fontdimen	Dimension name	Description
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.
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.

`\fontdimen`	Dimension name	Description
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.
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	RadicalDisplayStyleVerticalGap	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.

\fontdimen	Dimension name	Description
65	RadicalDegreeBottom-RaisePercent	Height of the bottom of the radical degree, if such is present, in proportion to the ascender of the radical sign. Suggested: 60%.

File IV
Some manner of unit testing

Some of the examples in the documentation are actually set up as unit tests, where multiple maths alphabets are placed on top of each other to ensure that various input methods result in the same output.

B The regular weight alphabets

For regular weight alphabets, we test the resolution from upright/italic math source to unified-shape output.

```
⟨*test⟩
\documentclass{article}
\usepackage[a6paper]{geometry}
\usepackage{fontspec}
\setmainfont{FPL Neu}
\usepackage{unicode-math}
\def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\def\uplatin{abcdefghijklmnopqrstuvwxyz}
\def\upGreek{ΑΒΓΔΕΖΗΘ□ΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}
\def\upgreek{αβγδε□ζηθ□ικ□λμνξοπ□ρ□ςστυφ□χψω}
\def\itLatin{□□□□□□□□□□□□□□□□□□□□□□□□□□}
\def\itlatin{□□□□□□□□□□□□□□□□□□□□□□□□□□}
\def\itGreek{□□□□□□□□□□□□□□□□□□□□□□□□□}
\def\itgreek{□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□}
\def\testmath#1{%
  \makebox[\linewidth][l]{%
    \makebox[0pt][l]{$\csname up#1\endcsname$}%
    \makebox[0pt][l]{$\csname it#1\endcsname$}}}
\begin{document}
\setmathfont[Colour=2255FF99]{Asana Math}
\parindent=0pt
\voffset=-1in
\hoffset=-1in
\setbox0=\vbox{%
\testmath{Latin}\\
\testmath{latin}\\
\testmath{Greek}\\
\testmath{greek}}
\dimen0=\ht0
\advance\dimen0\dp0
\edef\papersize{papersize=\the\wd0,\the\dimen0}
\setbox255=\vbox{\special{\papersize}\box0}
\shipout\box255
\end{document}
⟨/test⟩
```

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

C The bold alphabets

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

```
⟨*testbf⟩
\documentclass{article}
\usepackage[a6paper]{geometry}
\usepackage{fontspec}
\setmainfont{FPL Neu}
\usepackage{unicode-math}
\def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\def\uplatin{abcdefghijklmnopqrstuvwxyz}
\def\upGreek{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡ⍰ΣΤΥΦΧΨΩ}
\def\upgreek{αβγδεζηθικλμνξοπρςστυφχψω⍰⍰⍰⍰⍰⍰}
\def\itLatin{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\itlatin{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\itGreek{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\itgreek{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfupLatin{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfuplatin{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfupGreek{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfupgreek{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfitLatin{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfitlatin{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfitGreek{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\def\bfitgreek{⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰⍰}
\providecommand\mathalphabet{\mathbf}
\def\testmath#1{%
  \makebox[\linewidth][l]{%
```

```
    \makebox[0pt][l]{$\mathalphabet{\csname up#1\endcsname}$}%
    \makebox[0pt][l]{$\mathalphabet{\csname it#1\endcsname}$}%
    \makebox[0pt][l]{$\csname bfup#1\endcsname$}%
    \makebox[0pt][l]{$\csname bfit#1\endcsname$}%
    }}
\begin{document}
\setmathfont[Colour=2255FF55]{Asana Math}
\parindent=0pt
\voffset=-1in
\hoffset=-1in
\setbox0=\vbox{%
\testmath{Latin}\\
\testmath{latin}\\
\testmath{Greek}\\
\testmath{greek}}
\dimen0=\ht0
\advance\dimen0\dp0
\edef\papersize{papersize=\the\wd0,\the\dimen0}
\setbox255=\vbox{\special{\papersize}\box0}
\shipout\box255
\end{document}
⟨/testbf⟩
```

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.