Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/09/17   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 X∃TEX, although it is conjectured that some effect could be spent to create a cross-format package that would also work with LuaTEX.

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

2 Unicode maths font setup

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

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

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

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

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

> `\setmathfont[Version=Bold,`⟨*font features*⟩`]{`⟨*font name*⟩`}`

This has not been implemented yet.

Instances above of

[⟨*font features*⟩]{⟨*font name*⟩}

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

2.1 Using multiple fonts

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

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

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

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

2.2 Script and scriptscript fonts/features

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

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

3 Maths input

XƎTEX's unicode support allows maths input through two methods. Like classical TEX, macros such as \alpha, \sum, \pm, \leq, and so on, provide verbose access

to the entire repertoire of characters defined by unicode. The literal characters themselves may be used instead, for more readable input files.

3.1   Math 'style'

Classically, TeX uses italic lowercase Greek letters and *upright* uppercase Greek letters for variables in mathematics. This is contrary to the ISO standards of using italic forms for both upper- and lowercase. Furthermore, the French (contrary 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 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 the following page shows every character under the effect of this package option.

3.2   Bold style

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

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

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
αβγδεεζηθϑικϰλμνξοπϖρϱςστυφϕχψω

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
αβγδεεζηθϑικϰλμνξοπϖρϱςστυφϕχψω

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

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ
αβγδεεζηθϑικϰλμνξοπϖρϱςστυφϕχψω

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

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

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

Package option	Example	
	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`	$(\mathbf{a}, \mathbf{z}, \mathbf{B}, \mathbf{X})$	$(\mathbf{\alpha}, \mathbf{\beta}, \mathbf{\Gamma}, \mathbf{\Xi})$

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

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

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

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

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

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

3.3 Sans serif style

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

How should the generic \mathsf behave? Unlike bold, sans serif is used much more sparingly in mathematics. I've seen recommendations to typeset tensors in sans serif italic or sans serif italic bold (e.g., examples in the isomath and mattens packages). But LaTeX's \mathsf is *upright* sans serif.

Therefore I reluctantly add the package options [sans-style=TeX] and [sans-style=ISO] to control the behaviour of \mathsf. The TeX style sets up the command to use the seemingly-useless upright sans serif, including Greek; the ISO style switches to using italic in both Latin and Greek alphabets. In other words,

$$ABCDEFGHIJKLMNOPQRSTUVWXYZ$$
$$abcdefghijklmnopqrstuvwxyz$$
$$ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΘΣΤΥΦΧΨΩ$$
$$αβγδεζηθικλμνξοπρςστυφχψωϵϑϰφϱϖ$$

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

$$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$
$$\mathbf{abcdefghijklmnopqrstuvwxyz}$$
$$\mathbf{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΘΣΤΥΦΧΨΩ}$$
$$αβγδεζηθικλμνξοπρςστυφχψωϵϑϰφϱϖ$$

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

$$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$
$$\mathbf{abcdefghijklmnopqrstuvwxyz}$$
$$\mathbf{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΘΣΤΥΦΧΨΩ}$$
$$αβγδεζηθικλμνξοπρςστυφχψωϵϑϰφϱϖ$$

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

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

this option simply changes the meaning of \mathsf to either \mathsfup or \mathsfit, respectively. Please let me know if more granular control is necessary here.

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

3.3.1 What about bold sans serif?

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

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

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

3.4 All (the rest) of the mathematical alphabets

Unicode contains separate codepoints for most if not all variations of alphabet shape one may wish to use in mathematical notation. The complete list is shown

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

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

in table 3. Some of these have been covered in the previous sections.

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

3.5 Miscellanea

3.5.1 Nabla

The symbol ∇ comes in the six forms shown in table 4. We want an individual option to specify whether we want upright or italic nabla by default (when either upright or italic nabla is used in the source). TeX classically uses an upright nabla, but ɪso standards differ (I think). The package options nabla=upright and nabla=italic switch between the two choices. This is then inherited through \mathbf; \mathit and \mathup can be used to force one way or the other.

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

Table 4: The various forms of nabla.

Description		Glyph
Upright	Serif	∇
	Bold serif	**∇**
	Bold sans	⍰
Italic	Serif	∇
	Bold serif	**∇**
	Bold sans	⍰

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

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

3.5.2 Partial

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

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

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

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

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

TeX defines \epsilon to look like ε and \varepsilon to look like ϵ. The Unicode glyph directly after delta and before zeta is 'epsilon' and looks like ϵ; there is a subsequent variant of epsilon that looks like ε. This creates a problem. People

---

[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.

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

3.5.4 Primes

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

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

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

3.5.5 Unicode subscripts and superscripts

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

The range of subscripts and superscripts supported by this package are shown in figures 3 and 4. Please request more if you think it is appropriate.

$$\boxed{\mathrm{A}\ {}^{0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ +\ -\ =\ (\ )\ i\ n}\ \mathrm{Z}}$$

Figure 3: The unicode superscripts supported as input characters. These are the literal glyphs from Charis SIL, not the output seen when used for maths input. The 'A' and 'Z' are to provide context for the size and location of the superscript glyphs.

$$\boxed{\mathrm{A}\ {}_{0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ +\ -\ =\ (\ )\ a\ e\ i\ o\ r\ u\ v\ x\ \beta\ \gamma\ \rho\ \phi\ \chi}\ \mathrm{Z}}$$

Figure 4: The unicode subscripts supported as input characters. See note from figure 3.

3.5.6 Vertical bar '|'

3.5.7 Colon ':'

3.5.8 Slashes and backslashes

There are several slash-like symbols defined in unicode. The last two aren't defined by Cambria Math, so let's ignore them for now. I don't know what they're intended for.

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

3.5.9 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

Table 6: Slashes and backslashes.

Slot	Name	Glyph	Command	Intended (?) use
U+002F	SOLIDUS	/		text slash, not used for maths
U+2044	FRACTION SLASH	/	\fracslash	for creating 'vulgar' fractions
U+2215	DIVISION SLASH	/	\slash	to represent division without a built-up
U+29F8	BIG SOLIDUS	/	\xsol	who knows
U+005C	REVERSE SOLIDUS	\	\backslash	not for maths?
U+2216	SET MINUS	\	\smallsetminus	set operator
U+29F5	REVERSE SOLIDUS OPERATOR	\	\setminus	inverse division?
U+29F9	BIG REVERSE SOLIDUS	\	\xbsol	no idea

are the duplicates:

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

(Not yet implemented.)

File I
The unicode-math package

This is the package.

```
1 \ProvidesPackage{unicode-math}
2   [2009/09/17 v0.4 Unicode maths in XeLaTeX]
```

4   Things we need

Packages

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

Start using LaTeX3 — finally!

```
6 \ExplSyntaxOn
```

Counters and conditionals

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

For `math-style`:

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

For `bold-style`:

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

For `nabla`:

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

4.0.10 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]

```
23  \def\um@usv@num{`\0}
24  \def\um@usv@upLatin{`\A}
25  \def\um@usv@uplatin{`\a}
26  \def\um@usv@upGreek{"391}
27  \def\um@usv@upgreek{"3B1}
28  \def\um@usv@itLatin{"1D434}
29  \def\um@usv@itlatin{"1D44E}
30  \def\um@usv@itGreek{"1D6E2}
31  \def\um@usv@itgreek{"1D6FC}
32  \def\um@usv@bbnum{"1D7D8}
33  \def\um@usv@bbLatin{"1D538}
34  \def\um@usv@bblatin{"1D552}
35  \def\um@usv@scrLatin{"1D49C}
36  \def\um@usv@scrlatin{"1D4B6}
37  \def\um@usv@frakLatin{"1D504}
38  \def\um@usv@fraklatin{"1D51E}
39  \def\um@usv@sfnum{"1D7E2}
40  \def\um@usv@sfupLatin{"1D5A0}
41  \def\um@usv@sfLatin  {"1D5A0}
42  \def\um@usv@sfuplatin{"1D5BA}
43  \def\um@usv@sfitLatin{"1D608}
44  \def\um@usv@sfitlatin{"1D622}
```

---

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

13

```
45 \def\um@usv@ttnum{"1D7F6}
46 \def\um@usv@ttLatin{"1D670}
47 \def\um@usv@ttlatin{"1D68A}
```

Bold:

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

Greek variants:

```
72 \def\um@usv@varTheta{"3F4}
73 \def\um@usv@Digamma{"3DC}
74 \def\um@usv@varepsilon{"3F5}
75 \def\um@usv@vartheta{"3D1}
76 \def\um@usv@varkappa{"3F0}
77 \def\um@usv@varphi{"3D5}
78 \def\um@usv@varrho{"3F1}
79 \def\um@usv@varpi{"3D6}
80 \def\um@usv@digamma{"3DD}
```

Bold:

```
81 \def\um@usv@bfvarTheta{"1D6B9}
82 \def\um@usv@bfDigamma{"1D7CA}
83 \def\um@usv@bfvarepsilon{"1D6DC}
84 \def\um@usv@bfvartheta{"1D6DD}
85 \def\um@usv@bfvarkappa{"1D6DE}
86 \def\um@usv@bfvarphi{"1D6DF}
```

```
87  \def\um@usv@bfvarrho{"1D6E0}
88  \def\um@usv@bfvarpi{"1D6E1}
89  \def\um@usv@bfdigamma{"1D7CB}
```

Italic Greek variants:

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

Bold:

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

Nabla:

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

Partial:

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

4.1 Package options

xkeyval's package support is used here.

math-style

```
119  \define@choicekey*{unicode-math.sty}
```

15

```
120     {math-style}[\@tempa\@tempb]{iso,tex,french,literal}{
121  \ifcase\@tempb\relax
122    \@um@upGreekfalse
123    \@um@upgreekfalse
124    \@um@upLatinfalse
125    \@um@uplatinfalse
126    \@um@bfupGreekfalse
127    \@um@bfupgreekfalse
128    \@um@uppartialfalse
129    \@um@bfupLatinfalse
130    \@um@bfuplatinfalse
131    \@um@upNablafalse
132    \bool_set_false:N \g_um_upsans_bool
133    \bool_set_false:N \g_um_texgreek_bool
134  \or
135    \@um@upGreektrue
136    \@um@upgreekfalse
137    \@um@upLatinfalse
138    \@um@uplatinfalse
139    \@um@bfupGreektrue
140    \@um@bfupgreekfalse
141    \@um@uppartialfalse
142    \@um@bfupLatintrue
143    \@um@bfuplatintrue
144    \@um@upNablatrue
145    \bool_set_true:N \g_um_upsans_bool
146    \bool_set_true:N \g_um_texgreek_bool
147  \or
148    \@um@upGreektrue
149    \@um@upgreektrue
150    \@um@upLatintrue
151    \@um@uplatinfalse
152    \@um@bfupGreektrue
153    \@um@bfupgreektrue
154    \@um@uppartialtrue
155    \@um@bfupLatintrue
156    \@um@bfuplatintrue
157    \@um@upNablatrue
158    \bool_set_true:N \g_um_upsans_bool
159    \bool_set_false:N \g_um_texgreek_bool
160  \or
161    \@um@literaltrue
162    \@um@bfliteraltrue
163    \bool_set_true:N \g_um_sfliteral_bool
164    \bool_set_false:N \g_um_texgreek_bool
165  \fi
```

16

```
166 }
```

bold-style

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

sans-style

```
190 \bool_new:N \g_um_upsans_bool
191 \bool_new:N \g_um_sfliteral_bool
192 \define@choicekey*{unicode-math.sty}
193     {sans-style}[\@tempa\@tempb]{iso,tex,literal}{
194   \ifcase\@tempb\relax
195     \bool_set_false:N \g_um_upsans_bool
196   \or
197     \bool_set_true:N \g_um_upsans_bool
198   \or
199     \bool_set_true:N \g_um_sfliteral_bool
200   \fi
201 }
```

Symbol obliqueness

```
202 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
203   \ifcase\@tempb\relax
204     \@um@upNablatrue
```

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

Epsilon and phi shapes

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

4.2 Overcoming \@onlypreamble

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

```
247 \tl_map_inline:nn {
248 \new@mathgroup
249 \cdp@list
250 \cdp@elt
251 \DeclareMathSizes
252 \@DeclareMathSizes
253 \newmathalphabet
254 \newmathalphabet@@
255 \newmathalphabet@@@
256 \DeclareMathVersion
257 \define@mathalphabet
258 \define@mathgroup
259 \addtoversion
260 \version@list
261 \version@elt
262 \alpha@list
263 \alpha@elt
264 \restore@mathversion
265 \init@restore@version
266 \dorestore@version
267 \process@table
268 \new@mathversion
269 \DeclareSymbolFont
270 \group@list
271 \group@elt
272 \new@symbolfont
273 \SetSymbolFont
274 \SetSymbolFont@
275 \get@cdp
276 \DeclareMathAlphabet
277 \new@mathalphabet
278 \SetMathAlphabet
279 \SetMathAlphabet@
280 \DeclareMathAccent
281 \set@mathaccent
282 \DeclareMathSymbol
283 \set@mathchar
284 \set@mathsymbol
285 \DeclareMathDelimiter
286 \@xxDeclareMathDelimiter
287 \@DeclareMathDelimiter
288 \@xDeclareMathDelimiter
```

```
289  \set@mathdelimiter
290  \set@@mathdelimiter
291  \DeclareMathRadical
292  \mathchar@type
293  \DeclareSymbolFontAlphabet
294  \DeclareSymbolFontAlphabet@
295  }{
296    \tl_remove_in:Nn \@preamblecmds {\do#1}
297  }
```

4.3 Other things

\um@fontdimen@percent   #1 : Font dimen number
\fontdimens 10, 11, and 65 aren't actually dimensions, they're percentage values given in units of sp. This macro takes a font dimension number and outputs the decimal value of the associated parameter.

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

```
298  \def\um@fontdimen@percent#1#2{
299    0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax
300  }
```

\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.

```
301  \def\um@scaled@apply#1#2#3{
302    \ifx#1\scriptstyle
303      #2\um@fontdimen@percent{10}\um@font#3
304    \else
305      \ifx#1\scriptscriptstyle
306        #2\um@fontdimen@percent{11}\um@font#3
307      \else
308        #2#3%
309      \fi
310    \fi
311  }
```

5 Fundamentals

5.1 Enlarging the number of maths families

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

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

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

5.2 \DeclareMathSymbol **for unicode ranges**

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

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

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

The final macros that actually define the maths symbol with X∃TEX 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.

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

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

```
317   \ifx\mathop#3\relax
```

In order for literal unicode characters to be used in the source and still have the
correct limits behaviour, big operators are made math-active.

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

```
318     \begingroup
319       \char_make_active:n {#4}
320       \global\mathcode#4="8000\relax
321       \um@scanactivedef #4 \@nil { \csname\string#2@op\endcsname }
322     \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.

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

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

```
326    \cs_gset:cpn { \string#2 @op } {
327      \csname\string#2@sym\endcsname
328      \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
329      \ifin@
330        \expandafter\nolimits
331      \fi
332    }
```

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

```
333    \else
```

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

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

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

```
338    \ifx\mathopen#3\relax
339      \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
340      \global\XeTeXdelcode#4=#1 #4\relax
341      \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
342    \else
343      \ifx\mathclose#3\relax
344        \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
345        \global\XeTeXdelcode#4=#1 #4\relax
346        \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
347      \else
```

Accents

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

```
350        \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.

```
351            \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
352          \fi
353        \fi
354      \fi
355    \fi
356  \fi
357 }
```

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

```
358 \cs_set:Nn \um_set_mathcode:nnnn {
359   \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
360 }
```

5.3   The main `\setmathfont` macro

Here's the simplest usage:

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

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

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

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

```
\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

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

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

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

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

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

- Tell fontspec that maths font features are actually allowed.

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

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

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

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

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

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

```
369    \tl_set:Nn \l_um_script_features_tl  {ScriptStyle}
370    \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
371    \tl_set:Nn \l_um_script_font_tl       {#2}
372    \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.

```
373    \setkeys*[um]{options}{#1}
374    \edef\@tempa{\noexpand\zf@fontspec{
375        Script = Math,
376        SizeFeatures = {
377          {Size = \tf@size-} ,
378          {Size = \sf@size-\tf@size ,
379           Font = \l_um_script_font_tl ,
380           \l_um_script_features_tl
381          } ,
382          {Size = -\sf@size ,
383           Font = \l_um_sscript_font_tl ,
384           \l_um_sscript_features_tl
385          }
386        },
387        \XKV@rm
388     }{#2}
389    }
390    \@tempa
```

Probably want to check there that we're not creating multiple symbol fonts with the same NFSS declaration.

Check for the correct number of \fontdimens:

```
391  \font\um@font="#2"\relax
392  \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
393    \@um@ot@math@true
394  \else
395    \PackageWarningNoLine{unicode-math}{
396      The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
397      Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
398      in~ a~ substandard~ manner
399    }
400  \fi
```

If we're defining the full unicode math repetoire, then we skip all the parsing processing needed if we're only defining a subset.

- Math symbols are defined with \UnicodeMathSymbol; see section §5.3.1 for the individual definitions

```
401  \ifx\um@char@range\@empty
402    \tl_set:Nn \um_symfont_tl {um@allsym}
403    \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
404    \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
405    \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
406    \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
407    \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
408  \else
409    \stepcounter{um@fam}
410    \tl_set:Nx \um_symfont_tl {um@fam\theum@fam}
411    \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn
412    \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
413    \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
414    \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
415  \fi
```

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

```
416  \DeclareSymbolFont{\um_symfont_tl}
417    {\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.

```
418  \@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.)

```
419  \um_setup_shapes:
420  \um_remap_symbols:
421  \um_setup_mathactives:
422  \um_setup_alphanum:
423  \um_setup_alphabets:
```

End of the \setmathfont macro.

```
424  }
```

```
425  \cs_new:Nn \um_setup_shapes: {
426    \um_setup_nabla:
427    \um_setup_partial:
428  }
```

5.3.1 Functions for setting up symbols with mathcodes

\um_process_symbol_noparse:nnnn
\um_process_symbol_parse:nnnn

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

```
429  \cs_set:Nn \um_process_symbol_noparse:nnnn {
430    \um@mathsymbol{#2}{#3}{\um_symfont_tl}{#1}
431  }
432  \cs_set:Nn \um_process_symbol_parse:nnnn {
433    \um@parse@term{#1}{#2}{#3}{
434      \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
435    }
436  }
```

\um_remap_symbols:
\um_remap_symbol_noparse:nnn
\um_remap_symbol_parse:nnn

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

```
437  \cs_new:Nn \um_remap_symbols: {
438    \um_remap_symbol:nnn{`\-}{\mathbin}{"02212}% hyphen to minus
439    \um_remap_symbol:nnn{`\*}{\mathbin}{"02217}% text asterisk to "centred as-
    terisk"
440    \if@um@literal
441      \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
442      \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
443      \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
444      \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
445    \else
```

```
446    \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um_Nabla_up_or_it_usv}
447    \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um_partial_up_or_it_usv}
448    \fi
```

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

```
449    \if@um@bfliteral
450     \um_remap_symbol:nnn {\um@usv@bfNabla        }{\mathord}{\um@usv@bfNabla}
451    \um_remap_symbol:nnn {\um@usv@bfitNabla    }{\mathord}{\um@usv@bfitNabla}
452    \um_remap_symbol:nnn {\um@usv@bfsfNabla    }{\mathord}{\um@usv@bfsfNabla}
453    \um_remap_symbol:nnn {\um@usv@bfsfitNabla }{\mathord}{\um@usv@bfsfitNabla}
454     \um_remap_symbol:nnn {\um@usv@bfpartial    }{\mathord}{\um@usv@bfpartial}
455    \um_remap_symbol:nnn {\um@usv@bfitpartial }{\mathord}{\um@usv@bfitpartial}
456    \um_remap_symbol:nnn {\um@usv@bfsfpartial }{\mathord}{\um@usv@bfsfpartial}
457    \um_remap_symbol:nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
458    \else
459    \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla}{\mathord}{\um_bfNabla_up_or_it_usv}
460    \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla}{\mathord}{\um_bfsfNabla_up_or_it}
461    \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um_bfpartial_up_or_it}
462    \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um_bfsfpartial_u}
463    \fi
464 }
```

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

```
465 \cs_new:Nn \um_remap_symbol_parse:nnn {
466    \um@parse@term {#3} {\@nil} {#2} {
467      \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
468    }
469 }
470 \cs_new:Nn \um_remap_symbol_noparse:nnn {
471    \clist_map_inline:nn {#1} {
472      \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
473    }
474 }
```

5.3.2 Active math characters

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

`\um_setup_mathactives:`

```
475 \cs_new:Nn \um_setup_mathactives: {
476    \um_make_mathactive:nNN {"2032} \primesingle \mathord
477 }
```

`\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!

27

```
478  \cs_new:Nn \um_make_mathactive:nNN {
479    \XeTeXmathchardef #2 = "\mathchar@type #3
480                        \csname sym\um_symfont_tl\endcsname
481                        #1 \scan_stop:
482    \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:
483  }
```

5.3.3 Maths alphabets' character mapping

We want it to be convenient for users to actually type in maths. The ᴀꜱᴄɪɪ 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`.

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

Normal weight

```
487    \if@um@literal
488      \um_setup_literals:
489    \else
490      \um_setup_Latin:
491      \um_setup_latin:
492      \um_setup_Greek:
493      \um_setup_greek:
494    \fi
```

Bold

```
495    \if@um@bfliteral
496      \um_setup_bf_literals:
497    \else
498      \if@um@bfupLatin
499      \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfupLatin}
500      \else
501      \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
502      \fi
503      \if@um@bfuplatin
504      \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfuplatin}
505      \else
506      \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
507      \fi
508      \if@um@bfupGreek
509      \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfupGreek}
510      \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
```

28

```
511      \else
512      \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}
513      \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
514      \fi
515      \if@um@bfupgreek
516      \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
517      \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
518      \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
519      \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
520      \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
521      \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
522      \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
523      \else
524      \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
525      \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
526      \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
527      \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
528      \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
529      \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
530      \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
531      \fi
532      \fi
533    \else
```
: TODO : what is supposed to happen here?
```
534    \fi
535  }
```

5.3.4 Functions for setting up the maths alphabets

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

```
536  \cs_set:Nn \um_mathmap_noparse:Nnn {
537    \clist_map_inline:nn {#2} {
538      \exp_args:No \um@addto@mathmap \um_symfont_tl {##1}{#1}{#3}
539    }
540  }
```

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

When `\um@parse@term` is executed, it populates the `\um@char@num@range` macro with slot numbers corresponding to the specified range. This range is used to conditionally add `\um_set_mathcode:nnnn` declaractions to the maths alphabet definition (*e.g.*, `\um@mathscr`).

```
541 \cs_set:Nn \um_mathmap_parse:Nnn {
542   \clist_map_inline:Nn \um@char@num@range {
543     \ifnum##1=#3\relax
544       \clist_map_inline:nn {#2} {
545         \exp_args:No \um@addto@mathmap \um_symfont_tl {####1}{#1}{#3}
546       }
547     \fi
548   }
549 }
```

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

```
550 \newcommand\um@addto@mathmap[4]{
551   \expandafter\g@addto@macro
552     \csname um_setup_\cs_to_str:N #3:\endcsname{
553     \um_set_mathcode:nnnn{#2}{\mathalpha}{#1}{#4}
554   }
555 }
```

5.4 (Big) operators

Turns out that X$_{\exists}$TEX is clever enough to deal with big operators for us automatically with `\XeTeXmathchardef`. Amazing!

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

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

USV	Ex.	Macro	Description
U+02140	$\sum_{0}^{1}$	\Bbbsum	DOUBLE-STRUCK N-ARY SUMMATION

Code	Symbol	Command	Description
U+0220F	$\prod$	\prod	PRODUCT OPERATOR
U+02210	$\coprod$	\coprod	COPRODUCT OPERATOR
U+02211	$\sum$	\sum	SUMMATION OPERATOR
U+0222B	$\int$	\int	INTEGRAL OPERATOR
U+0222C	$\iint$	\iint	DOUBLE INTEGRAL OPERATOR
U+0222D	$\iiint$	\iiint	TRIPLE INTEGRAL OPERATOR
U+0222E	$\oint$	\oint	CONTOUR INTEGRAL OPERATOR
U+0222F	$\oiint$	\oiint	DOUBLE CONTOUR INTEGRAL OPERATOR
U+02230	$\oiiint$	\oiiint	TRIPLE CONTOUR INTEGRAL OPERATOR
U+02231	$\intclockwise$	\intclockwise	CLOCKWISE INTEGRAL
U+02232	$\varointclockwise$	\varointclockwise	CONTOUR INTEGRAL, CLOCKWISE
U+02233	$\ointctrclockwise$	\ointctrclockwise	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0	$\bigwedge$	\bigwedge	LOGICAL OR OPERATOR
U+022C1	$\bigvee$	\bigvee	LOGICAL AND OPERATOR
U+022C2	$\bigcap$	\bigcap	INTERSECTION OPERATOR
U+022C3	$\bigcup$	\bigcup	UNION OPERATOR
U+027D5	$\leftouterjoin$	\leftouterjoin	LEFT OUTER JOIN
U+027D6	$\rightouterjoin$	\rightouterjoin	RIGHT OUTER JOIN
U+027D7	$\fullouterjoin$	\fullouterjoin	FULL OUTER JOIN
U+027D8	$\bigbot$	\bigbot	LARGE UP TACK
U+027D9	$\bigtop$	\bigtop	LARGE DOWN TACK
U+029F8	$\xsol$	\xsol	BIG SOLIDUS
U+029F9	$\xbsol$	\xbsol	BIG REVERSE SOLIDUS
U+02A00	$\bigodot$	\bigodot	N-ARY CIRCLED DOT OPERATOR
U+02A01	$\bigoplus$	\bigoplus	N-ARY CIRCLED PLUS OPERATOR
U+02A02	$\bigotimes$	\bigotimes	N-ARY CIRCLED TIMES OPERATOR

31

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 \nolim-its suffix. This list is used when processing unicode-math-table.tex to define such commands automatically (see the macro \um@set@mathsymbol on page 20). 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 $\iiiint$, but that might be a matter of preference.

```
556 \def\um@nolimits{
557   \@elt\int\@elt\iint\@elt\iiint\@elt\iiiint\@elt\oint\@elt\oiint\@elt\oiiint
558   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
559   \@elt\intbar\@elt\intBar\@elt\fint\@elt\cirfnint\@elt\awint\@elt\rppolint
560   \@elt\scpolint\@elt\npolint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
561   \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
562 }
```

\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.

```
563 \newcommand\addnolimits[1]{
564   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
565 }
```

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

```
566 \def\removenolimits#1{
567   \begingroup
568     \def\@elt##1{
569       \ifx##1#1\else
570         \noexpand\@elt\noexpand##1
571       \fi}
572     \xdef\um@nolimits{\um@nolimits}
573   \endgroup
574 }
```

33

$$\iiint_V \quad \iiint \atop V \quad \iiint_V$$

5.5 Radicals

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

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

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

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

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

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

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

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

5.6 Delimiters

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

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

576 `\let\left@primitive\left`
577 `\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

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

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

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

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

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+0007C	|	\lvert	VERTICAL BAR
U+02016	‖	\lVert	DOUBLE VERTICAL BAR
U+0221A	√	\sqrt	RADICAL
U+0221B	∛	\cuberoot	CUBE ROOT
U+0221C	∜	\fourthroot	FOURTH ROOT
U+02308	⌈	\lceil	LEFT CEILING
U+0230A	⌊	\lfloor	LEFT FLOOR
U+0231C	⌜	\ulcorner	UPPER LEFT CORNER
U+0231E	⌞	\llcorner	LOWER LEFT CORNER
U+02772		\lbrbrak	LIGHT LEFT TORTOISE SHELL BRACKET ORNAMENT
U+027C5	⟅	\lbag	LEFT S-SHAPED BAG DELIMITER
U+027CC	⟌	\longdivision	LONG DIVISION
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

USV	Ex.	Macro	Description
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+0007C	|	\rvert	VERTICAL BAR
U+0007D	}	\rbrace	RIGHT CURLY BRACKET
U+02016	‖	\rVert	DOUBLE VERTICAL BAR
U+02309	⌉	\rceil	RIGHT CEILING
U+0230B	⌋	\rfloor	RIGHT FLOOR
U+0231D	⌝	\urcorner	UPPER RIGHT CORNER
U+0231F	⌟	\lrcorner	LOWER RIGHT CORNER
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

USV	Ex.	Macro	Description
U+029D9	⦙	\rvzigzag	RIGHT WIGGLY FENCE
U+029DB	⦛	\Rvzigzag	RIGHT DOUBLE WIGGLY FENCE
U+029FD	⟩	\rcurvyangle	RIGHT POINTING CURVED ANGLE BRACKET
U+03015		\rbrbrak	RIGHT BROKEN BRACKET
U+03019		\Rbrbrak	RIGHT WHITE TORTOISE SHELL BRACKET

5.7 Maths accents

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

USV	Ex.	Macro	Description
U+00300	$\grave{x}$	\grave	GRAVE ACCENT
U+00301	$\acute{x}$	\acute	ACUTE ACCENT
U+00302	$\hat{x}$	\hat	CIRCUMFLEX ACCENT
U+00303	$\tilde{x}$	\tilde	TILDE
U+00304	$\bar{x}$	\bar	MACRON
U+00305	$\overbar{x}$	\overbar	OVERBAR EMBELLISHMENT
U+00306	$\breve{x}$	\breve	BREVE
U+00307	$\dot{x}$	\dot	DOT ABOVE
U+00308	$\ddot{x}$	\ddot	DIERESIS
U+00309	$\ovhook{x}$	\ovhook	COMBINING HOOK ABOVE
U+0030A	$\ocirc{x}$	\ocirc	RING
U+0030C	$\check{x}$	\check	CARON
U+00310	$\candra{x}$	\candra	CANDRABINDU (NON-SPACING)
U+00312	x	\oturnedcomma	COMBINING TURNED COMMA ABOVE
U+00313	x	\osmooth	GREEK PSILI (SMOOTH BREATHING) (NON-SPACING)
U+00314	x	\orough	GREEK DASIA (ROUGH BREATHING) (NON-SPACING)
U+00315	x	\ocommatopright	COMBINING COMMA ABOVE RIGHT
U+0031A	x	\droang	LEFT ANGLE ABOVE (NON-SPACING)
U+020D0	x	\leftharpoonaccent	COMBINING LEFT HARPOON ABOVE
U+020D1	$\vec{x}$	\rightharpoonaccent	COMBINING RIGHT HARPOON ABOVE
U+020D2	x	\vertoverlay	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6	x	\overleftarrow	COMBINING LEFT ARROW ABOVE
U+020D7	$\vec{x}$	\overrightarrow	COMBINING RIGHT ARROW ABOVE
U+020DB	$\dddot{x}$	\dddot	COMBINING THREE DOTS ABOVE
U+020DC	$\ddddot{x}$	\ddddot	COMBINING FOUR DOTS ABOVE
U+020E1	x	\overleftrightarrow	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7		\annuity	COMBINING ANNUITY SYMBOL
U+020E8	x	\threeunderdot	COMBINING TRIPLE UNDERDOT
U+020E9	$\overline{x}$	\widebridgeabove	COMBINING WIDE BRIDGE ABOVE

U+020EC	☒	\underrightharpoondown	COMBINING RIGHTWARDS HARPOON WITH BARB DOWNWARDS
U+020ED	☒	\underleftharpoondown	COMBINING LEFTWARDS HARPOON WITH BARB DOWNWARDS
U+020EE	☒	\underleftarrow	COMBINING LEFT ARROW BELOW
U+020EF	☒	\underrightarrow	COMBINING RIGHT ARROW BELOW
U+020F0	☒	\asteraccent	COMBINING ASTERISK ABOVE

6 Font features

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

```
578 \newcommand\um@zf@feature[2]{
579   \define@key[zf]{options}{#1}[]{
580     \if@um@fontspec@feature
581       #2
582     \else
583       \PackageError{fontspec/unicode-math}
584         {The '#1' font feature can only be used for maths fonts}
585         {The feature you tried to use can only be in commands
586           like \protect\setmathfont}
587     \fi
588   }
589 }
```

6.1 OpenType maths font features

```
590 \um@zf@feature{ScriptStyle}{
591   \zf@update@ff{+ssty=0}
592 }
593 \um@zf@feature{ScriptScriptStyle}{
594   \zf@update@ff{+ssty=1}
595 }
```

6.2 Script and scriptscript font options

```
596 \define@cmdkey[um]{options}[um@]{ScriptFeatures}{}
597 \define@cmdkey[um]{options}[um@]{ScriptScriptFeatures}{}
598 \define@cmdkey[um]{options}[um@]{ScriptFont}{}
599 \define@cmdkey[um]{options}[um@]{ScriptScriptFont}{}
```

6.3 Range processing

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

```
600 \define@choicekey+[um]{options}{Range}[\@tempa\@tempb]{ALL}{
```

```
601    \ifcase\@tempb\relax
602      \global\let\um@char@range\@empty
603    \fi
604 }{
605    \xdef\um@char@range{#1}
606 }
```

Pretty basic comma separated range processing. Donald Arseneau's selectp package has a cleverer technique.

\um@parse@term   #1 : unicode character slot
#2 : control sequence (character macro)
#3 : control sequence (math type)
#4 : code to execute

This macro expands to #4 if any of its arguments are contained in the commalist \um@char@range. This list can contain either character ranges (for checking with #1) or control sequences. These latter can either be the command name of a specific character, *or* the math type of one (*e.g.,* \mathbin).

Character ranges are passed to \um@parse@range, which accepts input in the form shown in table 11.

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

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

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

```
607 \newcommand\um@parse@term[4]{
608    \clist_map_variable:NNn \um@char@range \@ii {
609      \unless\ifx\@ii\@empty
610        \@tempswafalse
```

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

```
611        \expandafter\um@firstchar\expandafter{\@ii}
612        \ifx\@tempa\um@backslash
613          \expandafter\ifx\@ii#2\relax
614            \@tempswatrue
615          \else
616            \expandafter\ifx\@ii#3\relax
617              \@tempswatrue
618            \fi
619          \fi
```

39

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

```
620        \else
621          \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
622        \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.

```
623        \if@tempswa
624          \ifx\um@char@num@range\@empty
625            \g@addto@macro\um@char@num@range{#1}
626          \else
627            \g@addto@macro\um@char@num@range{,#1}
628          \fi
629          #4%
630        \fi
631      \fi
632    }
633 }
634 \def\um@firstof#1#2\@nil{#1}
635 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
636 \def\um@firstchar#1{\edef\@tempa{\expandafter\um@firstof\string#1\@nil}}
```

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

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

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

```
637 \def\um@parse@range#1-#2-#3\@nil#4\@nil{
638    \def\@tempa{#1}
639    \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-{}

```
640    \expandafter\ifx\expandafter\@marker\@tempb\relax
641      \ifnum#4=#1\relax
642        \@tempswatrue
643      \fi
644    \else
```

40

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

```
645    \ifx\@empty\@tempb
646      \ifnum#4>\numexpr#1-1\relax
647        \@tempswatrue
648      \fi
649    \else
```

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

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

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

```
654      \else
655        \ifnum#4>\numexpr#1-1\relax
656          \ifnum#4<\numexpr#2+1\relax
657            \@tempswatrue
658          \fi
659        \fi
660      \fi
661    \fi
662  \fi
663 }
```

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

Loops through character ranges setting \mathcode.

```
664 \cs_set:Nn \um_map_chars_range:nnn {
665   \clist_map_variable:nNn {#2} \l_um_input_num {
666     \prg_stepwise_variable:nnnNn{0}{1}{#1} \l_um_incr_num {
667       \um_set_mathcode:nnnn
668         {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
669         {\mathalpha}{\um_symfont_tl}
670         {\numexpr \l_um_incr_num + #3 \relax}
671     }
672   }
```

41

```
673 }
674 \cs_set:Nn \um_map_chars_latin:nn {
675   \um_map_chars_range:nnn {25}{#1}{#2}
676 }
677 \cs_set:Nn \um_map_chars_greek:nn {
678   \um_map_chars_range:nnn {24}{#1}{#2}
679 }
680 \cs_set:Nn \um_map_chars_numbers:nn {
681   \um_map_chars_range:nnn {9}{#1}{#2}
682 }
683 \cs_set:Nn \um_map_char:nn {
684   \um_map_chars_range:nnn {0}{#1}{#2}
685 }
```

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

```
686 \cs_set:Npn \exp_args:Nnff {\::n\::f\::f\:::}
687 \cs_new:Nn \um_set_mathalphabet_char:Nnn {
688   \clist_map_variable:nNn {#2} \l_um_input_num {
689     \exp_args:Nnff \um_mathmap:Nnn {#1}
690       {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
691   }
692 }
```

\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.

```
693 \cs_new:Nn \um_set_mathalph_range:nNnn {
694   \clist_map_variable:nNn {#3} \l_um_input_num {
695     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
696       \exp_args:Nnff \um_mathmap:Nnn {#2}
697         {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
698         {\number\numexpr \l_um_inc_num + #4 \relax}
699     }
700   }
701 }
702 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
703   \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
704 }
705 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
706   \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
707 }
708 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
```

42

```
709     \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
710 }
```

BCDBCDEABCDEFG

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

6.4   Resolving Greek symbol name control sequences

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

```
711 \AtBeginDocument{\um@resolve@greek}
712 \newcommand\um@resolve@greek{
713     \def\Alpha{\mitAlpha}
714     \def\Beta{\mitBeta}
715     \def\Gamma{\mitGamma}
716     \def\Delta{\mitDelta}
717     \def\Epsilon{\mitEpsilon}
718     \def\Zeta{\mitZeta}
719     \def\Eta{\mitEta}
720     \def\Theta{\mitTheta}
721     \def\Iota{\mitIota}
722     \def\Kappa{\mitKappa}
723     \def\Lambda{\mitLambda}
724     \def\Mu{\mitMu}
725     \def\Nu{\mitNu}
726     \def\Xi{\mitXi}
727     \def\Omicron{\mitOmicron}
728     \def\Pi{\mitPi}
729     \def\Rho{\mitRho}
730     \def\varTheta{\mitvarTheta}
731     \def\Sigma{\mitSigma}
732     \def\Tau{\mitTau}
733     \def\Upsilon{\mitUpsilon}
734     \def\Phi{\mitPhi}
735     \def\Chi{\mitChi}
736     \def\Psi{\mitPsi}
737     \def\Omega{\mitOmega}
```

Lowercase:

```
738     \def\alpha{\mitalpha}
739     \def\beta{\mitbeta}
740     \def\gamma{\mitgamma}
```

43

```
741  \def\delta{\mitdelta}
742  \def\epsilon{
743    \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
744  }
745  \def\zeta{\mitzeta}
746  \def\eta{\miteta}
747  \def\theta{\mittheta}
748  \def\iota{\mitiota}
749  \def\kappa{\mitkappa}
750  \def\lambda{\mitlambda}
751  \def\mu{\mitmu}
752  \def\nu{\mitnu}
753  \def\xi{\mitxi}
754  \def\omicron{\mitomicron}
755  \def\pi{\mitpi}
756  \def\rho{\mitrho}
757  \def\varsigma{\mitvarsigma}
758  \def\sigma{\mitsigma}
759  \def\tau{\mittau}
760  \def\upsilon{\mitupsilon}
761  \def\phi{
762    \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}
763  }
764  \def\chi{\mitchi}
765  \def\psi{\mitpsi}
766  \def\omega{\mitomega}
767  \def\varepsilon{
768    \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
769  }
770  \def\vartheta{\mitvartheta}
771  \def\varkappa{\mitvarkappa}
772  \def\varphi{
773    \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
774  }
775  \def\varrho{\mitvarrho}
776  \def\varpi{\mitvarpi}
777  }
```

6.5 Setting up the mappings

\um_setup_literals: : TODO : other literal symbols

```
778  \cs_set:Nn \um_setup_literals: {
779    \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}
780    \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}
781    \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}
782    \um_map_char:nn {\um@usv@ith}{\um@usv@ith}
```

44

```
783    \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}
784    \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}
785    \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}
786    \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}
787    \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}
788  }
```

\um_setup_bf_literals: TODO: other literal symbols

```
789  \cs_set:Nn \um_setup_bf_literals: {
790    \um_map_chars_latin:nn {\um@usv@bfupLatin}{\um@usv@bfupLatin}
791    \um_map_chars_latin:nn {\um@usv@bfuplatin}{\um@usv@bfuplatin}
792    \um_map_chars_latin:nn {\um@usv@bfitLatin}{\um@usv@bfitLatin}
793    \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}
794    \um_map_chars_greek:nn {\um@usv@bfupGreek}{\um@usv@bfupGreek}
795    \um_map_chars_greek:nn {\um@usv@bfupgreek}{\um@usv@bfupgreek}
796    \um_map_chars_greek:nn {\um@usv@bfitGreek}{\um@usv@bfitGreek}
797    \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
798  }
```

\um_setup_Latin:

```
799  \cs_set:Nn \um_setup_Latin: {
800    \if@um@upLatin
801    \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
802    \else
803    \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
804    \fi
805  }
```

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

```
806  \cs_set:Nn \um_setup_latin: {
807    \if@um@uplatin
808    \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
809      \um_map_char:nn {\um@usv@ith}{`\h}
810    \else
811    \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
812      \um_map_char:nn {`\h,\um@usv@ith}{\um@usv@ith}
813    \fi
814  }
```

\um_setup_Greek:

```
815  \cs_set:Nn \um_setup_Greek: {
816    \if@um@upGreek
817    \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
818      \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}
819    \else
```

```
820    \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
821      \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}
822    \fi
823  }
```

\um_setup_greek:

```
824  \cs_set:Nn \um_setup_greek: {
825    \if@um@upgreek
826    \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
827    \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
828      \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
829      \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
830      \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
831      \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
832      \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
833    \else
834    \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
835    \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
836    \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
837    \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
838      \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
839      \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
840      \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
841    \fi
842  }
```

7   Maths alphabets mapping definitions

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

Implicit mode:

- Try and set all of the alphabet shapes.

- Check for the first glyph of the uppercase Latin alphabet to detect if
  the font supports each alphabet shape. (This doesn't work to distinguish
  Latin/Greek but we hope all maths fonts will have at least them!)

- For alphabets that do exist, overwrite whatever's already there.

- For alphabets that are not supported, *do nothing*. (This includes leaving the
  old alphabet definition in place.)

Explicit mode:

- Only set the alphabets specified.

- Check for the first glyph of the uppercase Latin alphabet to detect if the font contains the alphabet shape in the unicode math plane.

- For unicode math alphabets, overwrite whatever's already there.

- Otherwise, use the ASCII letters instead.

```
843 \cs_new:Nn \um_setup_alphabets: {
844   \um_setup_math_alphabet:nn {up    }{latin,Latin,greek,Greek}
845   \um_setup_math_alphabet:n {it    }
846   \um_setup_math_alphabet:n {bb    }
847   \um_setup_math_alphabet:nn {scr   }{latin,Latin}
848   \um_setup_math_alphabet:nn {frak  }{latin,Latin}
849   \um_setup_math_alphabet:n {sf    }
850   \um_setup_math_alphabet:n {sfup  }
851   \um_setup_math_alphabet:n {sfit  }
852   \um_setup_math_alphabet:n {tt    }
853   \um_setup_math_alphabet:n {bf    }
854   \um_setup_math_alphabet:n {bfup  }
855   \um_setup_math_alphabet:n {bfit  }
856   \um_setup_math_alphabet:n {bfscr }
857   \um_setup_math_alphabet:n {bffrak}
858   \um_setup_math_alphabet:n {bfsf  }
859   \um_setup_math_alphabet:n {bfsfup}
860   \um_setup_math_alphabet:n {bfsfit}
861 }
```

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

```
862 \cs_new:Nn \um_setup_math_alphabet:nn {
863   \clist_map_inline:nn {#2} {
864     \um_glyph_if_exist:nT {\csname um@usv@#1##1 \endcsname}{
865       \um_maybe_init_alphabet:n {#1}
866       \um_prepare_alph:n {#1}
867       \clist_map_break:
868     }
869   }
870   \clist_map_inline:nn {#2} {
871     \um_glyph_if_exist:nTF {\csname um@usv@#1##1 \endcsname}{
872       \use:c {um_config_math#1_##1:}
873     }{
874       \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
875       Math~ alphabet~
876       \@backslashchar math#1~
877       (\tl_use:c{g_um_math_alphabet_name_##1_tl})~
```

```
878      not~ found~ in~ font~
879      \fontname\um@font}
880    }
881  }
882 }
883 \tl_set:Nn \g_um_math_alphabet_name_latin_tl {Latin, lowercase}
884 \tl_set:Nn \g_um_math_alphabet_name_Latin_tl {Latin, uppercase}
885 \tl_set:Nn \g_um_math_alphabet_name_greek_tl {Greek, lowercase}
886 \tl_set:Nn \g_um_math_alphabet_name_Greek_tl {Greek, uppercase}
887 \tl_set:Nn \g_um_math_alphabet_name_num_tl   {Numerals}

888 \cs_new:Nn \um_setup_math_alphabet:n {
889   \um_glyph_if_exist:nTF {\csname um@usv@#1Latin \endcsname}{
890     \um_maybe_init_alphabet:n {#1}
891     \um_prepare_alph:n {#1}
892     \use:c {um_config_math#1:}
893   }{
894     \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
895     Math~ alphabet~ \@backslashchar math#1~ not~ found~ in~ font~ \font-
     name\um@font}
896     \cs_if_exist:cT {um_fix_math#1:} {
897       \use:c {um_fix_math#1:}
898     }
899   }
900 }
901 \cs_set:Nn \um_fix_mathtt: {
902  \SetMathAlphabet\mathtt{normal}\encodingdefault\ttdefault\mddefault\updefault
903 }
904 \cs_set:Nn \um_init_alphabet:n {
905   \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
906 }
```

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

```
907 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
908  \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
909 }
```

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

```
910 \cs_new:Nn \um_prepare_alph:n {
911   \cs_if_exist:cF {um_math#1:n} {
912     \cs_set:cpn {um_math#1:n} ##1 {
913       \use:c {um_setup_math#1:} ##1 \egroup
914     }
915     \cs_set_protected:cpn {math#1} {
```

48

```
916      \bgroup
917      \mode_if_math:F {
918        \egroup\expandafter
919        \non@alpherr\expandafter{\csname math#1\endcsname\space}
920      }
921      \use:c {um_math#1:n}
922    }
923  }
924 }
```

: TODO : nested alphabets?

7.1 Non-bold math alphabets

7.1.1 Upright: \mathup

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ   Θ
αβγδεζηθικλμνξοπρστυφχψω ϵϑϰϕϱϖ

```
$\mathup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathup{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathup{                          }$\quad$\mathup{ }$ \\
$\mathup{                          }$\quad$\mathup{      }$ \\
```

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

```
925 \cs_new:Npn \um_config_mathup_Latin: {
926   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
927 }
928 \cs_new:Npn \um_config_mathup_latin: {
929   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
930 }
931 \cs_new:Npn \um_config_mathup_Greek: {
932   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
933   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}
934   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varTheta}
935 }
936 \cs_new:Npn \um_config_mathup_greek: {
937   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
938   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}
939   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varep
940   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
941   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
942   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
943   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
944   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
945 }
```

7.1.2 Italic: `\mathit`

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
$abcdefghijklmnopqrstuvwxyz$
$ABΓΔEZHΘIKΛMNΞOΠPΣTΥΦXΨΩ$ $Θ$
$αβγδεζηθικλμνξοπρστυφχψω$ $ϵϑϰϕϱϖ$

```
$\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathit{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathit{                        }$\quad$\mathit{ }$ \\
$\mathit{                        }$\quad$\mathit{     }$ \\
```

Roman:

```
946  \cs_new:Npn \um_config_mathit: {
947    \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
948    \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
949    \um_set_mathalphabet_char:Nnn{\mathit}{`\h,\um@usv@ith}{\um@usv@ith}
```

Greek:

```
950    \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
951    \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
952    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
953    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartial}
954    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarThet
955    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvar
956    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvarthet
957    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkapp
958    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
959    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
960    \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
961  }
```

7.1.3 Blackboard or double-struck: `\mathbb`

0123456789
$ABCDEFGHIJKLMNOPQRSTUVWXYZ$
$abcdefghijklmnopqrstuvwxyz$

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

Numbers:

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

Roman uppercase:

```
964    \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
965    \um_set_mathalphabet_char:Nnn{\mathbb}{`\C,"1D60A}{"2102}
966    \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D60F}{"210D}
967    \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D60F}{"2115}
968    \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D617}{"2119}
969    \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D618}{"211A}
```

```
970  \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D619}{"211D}
971  \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D621} {"2124}
```

Roman lowercase:

```
972  \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}
973  }
```

7.1.4 Script or caligraphic: \mathscr and \mathcal

$\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
$\mathit{abcdefghijklmnopqrstuvwxyz}$

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

```
974  \cs_new:Npn \um_config_mathscr_Latin: {
975  \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}
976  \um_set_mathalphabet_char:Nnn  \mathscr {`\B,"1D435}{"212C}
977  \um_set_mathalphabet_char:Nnn  \mathscr {`\E,"1D438}{"2130}
978  \um_set_mathalphabet_char:Nnn  \mathscr {`\F,"1D439}{"2131}
979  \um_set_mathalphabet_char:Nnn  \mathscr {`\H,"1D43B}{"210B}
980  \um_set_mathalphabet_char:Nnn  \mathscr {`\I,"1D43C}{"2110}
981  \um_set_mathalphabet_char:Nnn  \mathscr {`\L,"1D43F}{"2112}
982  \um_set_mathalphabet_char:Nnn  \mathscr {`\M,"1D440}{"2133}
983  \um_set_mathalphabet_char:Nnn  \mathscr {`\R,"1D445}{"211B}
984  }
985  \cs_new:Npn \um_config_mathscr_latin: {
986  \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrlatin}
987  \um_set_mathalphabet_char:Nnn  \mathscr {`\e,"1D452}{"212F}
988  \um_set_mathalphabet_char:Nnn  \mathscr {`\g,"1D454}{"210A}
989  \um_set_mathalphabet_char:Nnn  \mathscr {`\o,"1D45C}{"2134}
990  }
```

7.1.5 Fractur or fraktur or blackletter: \mathfrak

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

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

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

```
991  \cs_new:Npn \um_config_mathfrak_Latin: {
992  \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLatin}
993  \um_set_mathalphabet_char:Nnn  \mathfrak {`\C,"1D436}{"212D}
994  \um_set_mathalphabet_char:Nnn  \mathfrak {`\H,"1D43B}{"210C}
995  \um_set_mathalphabet_char:Nnn  \mathfrak {`\I,"1D43C}{"2111}
996  \um_set_mathalphabet_char:Nnn  \mathfrak {`\R,"1D445}{"211C}
997  \um_set_mathalphabet_char:Nnn  \mathfrak {`\Z,"1D44D}{"2128}
```

```
998 }
999 \cs_new:Npn \um_config_mathfrak_latin: {
1000   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklatin}
1001 }
```

7.1.6 Sans serif: \mathsf

<div style="text-align:center">

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

</div>

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

```
1002 \cs_new:Npn \um_config_mathsf: {
1003   \bool_if:NTF \g_um_sfliteral_bool {
1004     \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnum}
1005   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin}{\um@usv@sfupLatin}
1006   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin}{\um@usv@sfuplatin}
1007   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itLatin}{\um@usv@sfitLatin}
1008   \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itlatin}{\um@usv@sfitlatin}
1009   }{
1010     \bool_if:NTF \g_um_upsans_bool {
1011       \um_set_mathalphabet_numbers:Nnn \mathsf {\um@usv@num}{\um@usv@sfnum}
1012     \um_set_mathalphabet_latin:Nnn  \mathsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfupLati
1013     \um_set_mathalphabet_latin:Nnn  \mathsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfuplati
1014     }{
1015       \um_set_mathalphabet_numbers:Nnn \mathsf {\um@usv@num}{\um@usv@sfnum}
1016     \um_set_mathalphabet_latin:Nnn  \mathsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLati
1017     \um_set_mathalphabet_latin:Nnn  \mathsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlati
1018     }
1019   }
1020 }
```

7.1.7 Sans serif upright: \mathsfup

<div style="text-align:center">

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz

</div>

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

```
1021 \cs_new:Npn \um_config_mathsfup: {
1022   \um_set_mathalphabet_numbers:Nnn{\mathsfup}{\um@usv@num}{\um@usv@sfnum}
1023   \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfupLatin]
1024   \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfuplatin]
1025 }
```

7.1.8 Sans serif italic: \mathsfit

$$0123456789$$
$$\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$
$$\mathit{abcdefghijklmnopqrstuvwxyz}$$

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

```
1026  \cs_new:Npn \um_config_mathsfit: {
1027    \um_set_mathalphabet_numbers:Nnn{\mathsfit}{\um@usv@num}{\um@usv@sfnum}
1028    \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLatin}
1029    \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlatin}
1030  }
```

7.1.9 Typewriter or monospaced: \mathtt

```
0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
```

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

```
1031  \cs_new:Npn \um_config_mathtt: {
1032    \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
1033    \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
1034    \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlatin}
1035  }
```

7.2 Bold math alphabets

7.2.1 Bold: \mathbf

$$\mathbf{0123456789}$$
$$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$
$$\mathbf{abcdefghijklmnopqrstuvwxyz}$$
$$\mathbf{ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ}$$
$$\mathbf{Θ⸮}$$
$$\mathbf{αβγδεζηθικλμνξοπρστυφχψω}$$
$$\mathbf{ϵϑϰϕϱϖ⸮}$$

```
$\mathbf{0123456789}$ \\
$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbf{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbf{                          }$\quad$\mathbf{   }$ \\
$\mathbf{                          }$\quad$\mathbf{         }$ \\
```

```
1036  \cs_new:Npn \um_config_mathbf: {
1037    \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
1038    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{"1D7CA}
1039    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{"1D7CB}
1040    \if@um@bfliteral
```

```
1041    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfupLatin}
1042    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
1043    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@bfuplatin}
1044    \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@bfitlatin}
1045    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfupGreek}
1046    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
1047    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfupgreek}
1048    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitgreek}
1049     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1050    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
1051     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
1052    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
1053    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
1054    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bfvarepsilon}
1055    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
1056    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
1057    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
1058    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
1059     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}
1060    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
1061    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
1062    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
1063    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
1064    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
1065    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
1066    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1067    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
1068    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
1069    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
1070   \else
1071     \if@um@bfupLatin
1072     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLatin
1073     \else
1074     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin
1075     \fi
1076     \if@um@bfuplatin
1077     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplatin
1078       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
1079     \else
1080     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin
1081       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1082     \fi
1083     \if@um@bfupGreek
1084     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGreek
1085     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarT
1086     \else
```

```
1087    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek
1088    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitva
1089    \fi
1090    \if@um@bfupgreek
1091    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgreek
1092    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf
1093    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvart
1094    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvark
1095    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1096    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1097    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1098    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartia
1099    \else
1100    \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek
1101    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bf
1102    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitva
1103    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitva
1104    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi
1105    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho
1106    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1107    \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpart
1108    \fi
1109  \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um_bfNabla_up_or_it_
1110  \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um_bfpartial_up_
1111  \fi
1112 }
```

7.2.2 Bold Italic: \mathbfit

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ   Θ
αβγδεζηθικλμνξοπρστυφχψω   εϑϰϕϱϖ

```
$\mathbfit{0123456789}$ \\
$\mathbfit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfit{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbfit{                         }$\quad
  $\mathbfit{ }$ \\
$\mathbfit{                         }$\quad
  $\mathbfit{        }$ \\
```

```
1113 \cs_new:Npn \um_config_mathbfit: {
1114   \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnum}
1115   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin
1116   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin
1117   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek
1118   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek
1119   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfupLatin}{\um@usv@bfitLatin}
1120   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@bfuplatin}{\um@usv@bfitlatin}
1121   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfupGreek}{\um@usv@bfitGreek}
```

```
1122    \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@bfupgreek}{\um@usv@bfitgreek}
1123    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitvar
1124    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}
1125    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitparti
1126    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfi
1127    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitvar
1128    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitvar
1129    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi}
1130    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho}
1131    \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1132  }
```

7.2.3   Bold Italic: \mathbfup

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ   Θ
αβγδεζηθικλμνξοπρστυφχψω   ϵϑϰφϱϖ

```
$\mathbfup{0123456789}$ \\
$\mathbfup{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ \\
$\mathbfup{abcdefghijklmnopqrstuvwxyz}$ \\
$\mathbfup{                      }$\quad
  $\mathbfup{ }$ \\
$\mathbfup{                          }$\quad
  $\mathbfup{        }$ \\
```

```
1133  \cs_new:Npn \um_config_mathbfup: {
1134    \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnum}
1135    \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLatin
1136    \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplatin
1137    \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGreek
1138    \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgreek
1139    \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bfupLatin}{\um@usv@bfupLatin}
1140    \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@bfuplatin}{\um@usv@bfuplatin}
1141    \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfupGreek}{\um@usv@bfupGreek}
1142    \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@bfupgreek}{\um@usv@bfupgreek}
1143    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfvarTh
1144    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}
1145    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpartial
1146    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfv
1147    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvarth
1148    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarka
1149    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1150    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1151    \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1152  }
```

7.2.4 Bold fractur or fraktur or blackletter: \mathbffrak

𝕬𝕭𝕮𝕯𝕰𝕱𝕲𝕳𝕴𝕵𝕶𝕷𝕸𝕹𝕺𝕻𝕼𝕽𝕾𝕿𝖀𝖁𝖂𝖃𝖄𝖅
𝖆𝖇𝖈𝖉𝖊𝖋𝖌𝖍𝖎𝖏𝖐𝖑𝖒𝖓𝖔𝖕𝖖𝖗𝖘𝖙𝖚𝖛𝖜𝖝𝖞𝖟

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

```
1153 \cs_new:Npn \um_config_mathbffrak: {
1154    \um_set_mathalphabet_numbers:Nnn{\mathbffrak}{\um@usv@num}{\um@usv@bfnum}
1155    \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin, \um@usv@itLatin,\um@usv@frakLati
1156    \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin,\um@usv@fraklatin
1157 }
```

7.2.5 Bold script or calligraphic: \mathbfscr

𝓐𝓑𝓒𝓓𝓔𝓕𝓖𝓗𝓘𝓙𝓚𝓛𝓜𝓝𝓞𝓟𝓠𝓡𝓢𝓣𝓤𝓥𝓦𝓧𝓨𝓩
𝓪𝓫𝓬𝓭𝓮𝓯𝓰𝓱𝓲𝓳𝓴𝓵𝓶𝓷𝓸𝓹𝓺𝓻𝓼𝓽𝓾𝓿𝔀𝔁𝔂𝔃

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

```
1158 \cs_new:Npn \um_config_mathbfscr: {
1159    \um_set_mathalphabet_numbers:Nnn{\mathbfscr}{\um@usv@num}{\um@usv@bfnum}
1160    \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrLati
1161    \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrlati
1162 }
```

7.2.6 Bold sans serif: \mathbfsf

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ   Θ
αβγδεζηθικλμνξοπρστυφχψω   ϵϑϰϕϱϖ

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

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

```
1163 \cs_new:Npn \um_config_mathbfsf: {
1164    \bool_if:NTF \g_um_sfliteral_bool {
1165    \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1166    \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@upLatin}{\um@usv@bfsfupLatin}
1167    \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@uplatin}{\um@usv@bfsfuplatin}
1168    \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@itLatin}{\um@usv@bfsfitLatin}
1169    \um_set_mathalphabet_latin:Nnn  \mathbfsf {\um@usv@itlatin}{\um@usv@bfsfitlatin}
1170    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@upGreek}{\um@usv@bfsfupGreek}
```

```
1171    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@upgreek}{\um@usv@bfsfupgreek}
1172    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@itGreek}{\um@usv@bfsfitGreek}
1173    \um_set_mathalphabet_greek:Nnn  \mathbfsf {\um@usv@itgreek}{\um@usv@bfsfitgreek}
1174     \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@varTheta}{"1D767}
1175     \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@Nabla}{"1D76F}
1176     \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@partial}{"1D789}
1177    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varepsilon}{"1D78A}
1178    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@vartheta}{"1D78B}
1179    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varkappa}{"1D78C}
1180    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varphi}{"1D78D}
1181    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varrho}{"1D78E}
1182    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varpi}{"1D78F}
1183    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varTheta}{"1D7A1}
1184   \um_set_mathalphabet_char:Nnn  \mathbfsf {\um@usv@itNabla}{\um@usv@bfsfitNabla}
1185   \um_set_mathalphabet_char:Nnn  \mathbfsf {\um@usv@itpartial}{\um@usv@bfsfitpartial}
1186    \um_set_mathalphabet_char:Nnn   \mathbfsf {\um@usv@itvarepsilon}{"1D7C4}
1187     \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvartheta}{"1D7C5}
1188     \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvarkappa}{"1D7C6}
1189     \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvarphi}{"1D7C7}
1190     \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvarrho}{"1D7C8}
1191     \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@itvarpi}{"1D7C9}
1192    }{
1193      \bool_if:NTF \g_um_upsans_bool {
1194      \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1195      \um_set_mathalphabet_latin:Nnn   \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfup
1196      \um_set_mathalphabet_latin:Nnn   \mathbfsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfup
1197      \um_set_mathalphabet_greek:Nnn   \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfup
1198      \um_set_mathalphabet_greek:Nnn   \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfup
1199      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{"1D767}
1200      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{"1D76F}
1201      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@partial,\um@usv@itpartial}{"1D789}
1202      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D78A}
1203      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{"1D78B}
1204      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{"1D78C}
1205      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{"1D78D}
1206      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{"1D78E}
1207      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{"1D78F}
1208      }{
1209      \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1210      \um_set_mathalphabet_latin:Nnn   \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfit
1211      \um_set_mathalphabet_latin:Nnn   \mathbfsf {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfit
1212      \um_set_mathalphabet_greek:Nnn   \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfit
1213      \um_set_mathalphabet_greek:Nnn   \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfit
1214         \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varTheta}{"1D7A1}
1215      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNab
1216      \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsfi
```

```
1217    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D7C4}
1218    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{"1D7C5}
1219    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{"1D7C6}
1220    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{"1D7C7}
1221    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{"1D7C8}
1222    \um_set_mathalphabet_char:Nnn    \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{"1D7C9}    }
1223  }
1224 }
```

7.2.7 Bold upright sans serif: `\mathbfsfup`

<div>

0123456789
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ   Θ
αβγδεζηθικλμνξοπρστυφχψω   εϑϰφϱϖ

</div>

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

Numbers (always upright) and letters:

```
1225 \cs_new:Npn \um_config_mathbfsfup: {
1226  \um_set_mathalphabet_numbers:Nnn{\mathbfsfup}{\um@usv@num}{\um@usv@bfsfnum}
1227  \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfupLa
1228  \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfupla
1229  \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfupGr
1230  \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfupgr
```

Others:

```
1231  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varTheta,\um@usv@itvarTheta}{"1D767}
1232  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@Nabla,\um@usv@itNabla}{"1D76F}
1233  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@partial,\um@usv@itpartial}{"1D789}
1234  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D78A}
1235  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@vartheta,\um@usv@itvartheta}{"1D78B}
1236  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varkappa,\um@usv@itvarkappa}{"1D78C}
1237  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varphi,\um@usv@itvarphi}{"1D78D}
1238  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varrho,\um@usv@itvarrho}{"1D78E}
1239  \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varpi,\um@usv@itvarpi}{"1D78F}
1240 }
```

7.2.8 Bold italic sans serif: `\mathbfsfit`

$ABCDEFGHIJKLMNOPQRSTUVWXYZ$

$abcdefghijklmnopqrstuvwxyz$

$ABΓΔEZHΘIKΛMNΞOΠPΣTYΦXΨΩ \quad Θ$

$αβγδεζηθικλμνξοπρστυφχψω \quad ϵϑϰϕϱϖ$

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

```
1241 \cs_new:Npn \um_config_mathbfsfit: {
1242   \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@num}{\um@usv@bfsfnum}
1243   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfitLa
1244   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfitla
1245   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfitGr
1246   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfitgr
```

Other symbols:

```
1247   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{"1D7A1}
1248   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNabla
1249   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsfitp
1250   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{"1D7C4}
1251   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{"1D7C5}
1252   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{"1D7C6}
1253   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{"1D7C7}
1254   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{"1D7C8}
1255   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{"1D7C9}
1256 }
```

7.3 Definitions of the math symbols

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

`\um@scancharlet`
`\um@scanactivedef`

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

```
1257 \begingroup
1258   \char_make_other:N \^
1259   \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1260     \lowercase{
1261       \scantokens{\global\let#1=^^^^^#2}
```

```
1262        }
1263      }
```

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

```
1264    \gdef\um@scanactivedef"#1\@nil#2{
1265      \lowercase{
1266        \tl_rescan:nn{
1267          \char_make_math_superscript:N\^
1268        }{
1269          \global\def^^^^^#1{#2}
1270        }
1271      }
1272    }
1273  \endgroup
```

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

```
1274  \begingroup
1275    \def\UnicodeMathSymbol#1#2#3#4{
1276      \um@scancharlet#2=#1\@nil
1277    }
1278    \char_make_other:N \#
1279    \@input{unicode-math-table.tex}
1280  \endgroup
```

8 Epilogue

Lots of little things to tidy up.

8.0.1 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 super-scripted. However, in a correctly behaviour OpenType font with the MATH table, we also see different behaviour after the ssty feature is applied:

U+2032: PRIME in the 'scriptstyle' font: x′

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

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

- Prime encountered; pcount=1.

- Scan ahead; if prime: pcount:=pcount+1; repeat.

- If not prime, stop scanning.

- If pcount=1, \prime, end.

- If pcount=2, check \primedouble; if it exists, use it, end; if not, goto last step.

- Ditto pcount=3 & \primetriple.

- Ditto pcount=4 & \primequadruple.

- If pcount>4 or the glyph doesn't exist, insert pcount \primes with \primekern between each.

```
1281 \muskip_new:N \g_um_primekern_muskip
1282 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1283 \num_new:N \l_um_primecount_num
1284 \cs_new:Nn \um_nprimes:n {
1285   ^{
1286     \primesingle
1287     \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1288   }
1289 }
1290 \cs_new:Nn \um_nprimes_select:n {
1291   \prg_case_int:nnn {#1}{
1292     {1} { ^{\primesingle} }
1293     {2} {
1294     \um_glyph_if_exist:nTF {"2033} { ^{\primedouble} } {\um_nprimes:n {#1}}
1295     }
1296     {3} {
```

```
1297        \um_glyph_if_exist:nTF {"2034} {^{\primetriple} } {\um_nprimes:n {#1}}
1298      }
1299      {4} {
1300      \um_glyph_if_exist:nTF {"2057} { ^{\primequadruple} } {\um_nprimes:n {#1}}
1301      }
1302  }{
1303    \um_nprimes:n {#1}
1304  }
1305 }
```

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.

```
1306 \cs_new:Nn \um_scanprime: {
1307    \num_zero:N \l_um_primecount_num
1308    \um_scanprime_collect:
1309 }
1310 \cs_new:Nn \um_scanprime_collect: {
1311    \num_incr:N \l_um_primecount_num
1312    \peek_meaning_remove:NTF ' {
1313      \um_scanprime_collect:
1314    }{
1315      \peek_meaning_remove:NTF \um_scanprime: {
1316        \um_scanprime_collect:
1317      }{
1318        \peek_meaning_remove:NTF ^^^^2032 {
1319          \um_scanprime_collect:
1320        }{
1321          \um_nprimes_select:n {\l_um_primecount_num}
1322        }
1323      }
1324    }
1325 }
1326 \cs_set_eq:NN \prime \um_scanprime:
1327 \group_begin:
1328   \char_make_active:N \'
1329   \char_make_active:n {"2032}
1330   \cs_gset_eq:NN ' \um_scanprime:
1331   \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1332 \group_end:
```

8.0.2 Unicode radicals

Undo the damage made to \sqrt:

```
1333 \DeclareRobustCommand\sqrt{\@ifnextchar[\@sqrt\sqrtsign}
```

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

   #2 : Leading superscript for the sqrt sign

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

```
1334 \def\r@@t#1#2{
1335     \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
1336     \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1337     \raise \dimexpr(
1338         \um@fontdimen@percent{65}{\um@font}\ht\z@-
1339         \um@fontdimen@percent{65}{\um@font}\dp\z@
1340       )\relax
1341       \copy \rootbox
1342     \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1343     \box \z@
1344 }
```

8.0.3 Unicode sub- and super-scripts

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

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

First, the setup of each mathactive char:

```
1345 \prop_new:N \g_um_supers_prop
1346 \prop_new:N \g_um_subs_prop
1347 \cs_generate_variant:Nn \prop_gput:Nnn {Nxn}
1348 \cs_generate_variant:Nn \prop_get:NnN {cxN}
1349 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}
1350
1351 \group_begin:
1352
1353 % Populate a property list with superscript characters; their mean-
     ing as their key,
1354 % for reasons that will become apparent soon, and their replace-
     ment as each key's value.
1355 % Then make the superscript active and bind it to the scanning function.
1356 %
1357 % \cs{scantokens} makes this process much simpler since we can acti-
     vate the char
1358 % and assign its meaning in one step.
1359 \cs_set:Nn \um_setup_active_superscript:nn {
```

```
1360    \prop_gput:Nxn \g_um_supers_prop    {\meaning #1} {#2}
1361    \char_make_active:n {`#1}
1362    \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1363    \scantokens{
1364      \cs_gset:Npn #1 {
1365        \tl_set:Nn \l_um_ss_chain_tl {#2}
1366        \cs_set_eq:NN \um_sub_or_super:n \sp
1367        \tl_set:Nn \l_um_tmpa_tl {supers}
1368        \um_scan_sscript:
1369      }
1370    }
1371  }
1372
1373  \um_setup_active_superscript:nn {^^^^2070} {0}
1374  \um_setup_active_superscript:nn {^^^^00b9} {1}
1375  \um_setup_active_superscript:nn {^^^^00b2} {2}
1376  \um_setup_active_superscript:nn {^^^^00b3} {3}
1377  \um_setup_active_superscript:nn {^^^^2074} {4}
1378  \um_setup_active_superscript:nn {^^^^2075} {5}
1379  \um_setup_active_superscript:nn {^^^^2076} {6}
1380  \um_setup_active_superscript:nn {^^^^2077} {7}
1381  \um_setup_active_superscript:nn {^^^^2078} {8}
1382  \um_setup_active_superscript:nn {^^^^2079} {9}
1383  \um_setup_active_superscript:nn {^^^^207a} {+}
1384  \um_setup_active_superscript:nn {^^^^207b} {-}
1385  \um_setup_active_superscript:nn {^^^^207c} {=}
1386  \um_setup_active_superscript:nn {^^^^207d} {(}
1387  \um_setup_active_superscript:nn {^^^^207e} {)}
1388  \um_setup_active_superscript:nn {^^^^2071} {i}
1389  \um_setup_active_superscript:nn {^^^^207f} {n}
1390
1391  % Ditto above.
1392  \cs_set:Nn \um_setup_active_subscript:nn {
1393    \prop_gput:Nxn \g_um_subs_prop    {\meaning #1} {#2}
1394    \char_make_active:n {`#1}
1395    \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1396    \scantokens{
1397      \cs_gset:Npn #1 {
1398        \tl_set:Nn \l_um_ss_chain_tl {#2}
1399        \cs_set_eq:NN \um_sub_or_super:n \sb
1400        \tl_set:Nn \l_um_tmpa_tl {subs}
1401        \um_scan_sscript:
1402      }
1403    }
1404  }
1405
```

```
1406  \um_setup_active_subscript:nn {^^^^2080} {0}
1407  \um_setup_active_subscript:nn {^^^^2081} {1}
1408  \um_setup_active_subscript:nn {^^^^2082} {2}
1409  \um_setup_active_subscript:nn {^^^^2083} {3}
1410  \um_setup_active_subscript:nn {^^^^2084} {4}
1411  \um_setup_active_subscript:nn {^^^^2085} {5}
1412  \um_setup_active_subscript:nn {^^^^2086} {6}
1413  \um_setup_active_subscript:nn {^^^^2087} {7}
1414  \um_setup_active_subscript:nn {^^^^2088} {8}
1415  \um_setup_active_subscript:nn {^^^^2089} {9}
1416  \um_setup_active_subscript:nn {^^^^208a} {+}
1417  \um_setup_active_subscript:nn {^^^^208b} {-}
1418  \um_setup_active_subscript:nn {^^^^208c} {=}
1419  \um_setup_active_subscript:nn {^^^^208d} {(}
1420  \um_setup_active_subscript:nn {^^^^208e} {)}
1421  \um_setup_active_subscript:nn {^^^^2090} {a}
1422  \um_setup_active_subscript:nn {^^^^2091} {e}
1423  \um_setup_active_subscript:nn {^^^^1d62} {i}
1424  \um_setup_active_subscript:nn {^^^^2092} {o}
1425  \um_setup_active_subscript:nn {^^^^1d63} {r}
1426  \um_setup_active_subscript:nn {^^^^1d64} {u}
1427  \um_setup_active_subscript:nn {^^^^1d65} {v}
1428  \um_setup_active_subscript:nn {^^^^2093} {x}
1429  \um_setup_active_subscript:nn {^^^^1d66} {\beta}
1430  \um_setup_active_subscript:nn {^^^^1d67} {\gamma}
1431  \um_setup_active_subscript:nn {^^^^1d68} {\rho}
1432  \um_setup_active_subscript:nn {^^^^1d69} {\phi}
1433  \um_setup_active_subscript:nn {^^^^1d6a} {\chi}
1434
1435  \group_end:
1436
1437  % The scanning command, evident in its purpose:
1438  \cs_new:Nn \um_scan_sscript: {
1439    \um_scan_sscript:TF {
1440      \um_scan_sscript:
1441    }{
1442      \um_sub_or_super:n {\l_um_ss_chain_tl}
1443    }
1444  }
1445
1446  % The main theme here is stolen from the source to the various \cs{peek_} func-
     tions.
1447  % Consider this function as simply boilerplate:
1448  \cs_new:Nn \um_scan_sscript:TF {
1449    \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
1450    \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
```

```
1451    \tl_set:Nx \l_peek_false_tl {\exp_not:n{\group_align_safe_end: #2}}
1452    \group_align_safe_begin:
1453      \peek_after:NN \um_peek_execute_branches_ss:
1454  }
1455
1456  % We do not skip spaces when scanning ahead, and we explicitly wish to
1457  % bail out on encountering a space or a brace.
1458  \cs_new:Npn \um_peek_execute_branches_ss: {
1459    \bool_if:nTF {
1460      \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
1461      \token_if_eq_catcode_p:NN \l_peek_token \c_group_end_token ||
1462      \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
1463    }
1464    { \l_peek_false_tl  }
1465    { \um_peek_execute_branches_ss_aux: }
1466  }
1467
1468  % This is the actual comparison code.
1469  % Because the peeking has already tokenised the next token,
1470  % it's too late to extract its charcode directly. Instead,
1471  % we look at its meaning, which remains a `character' even
1472  % though it is itself math-active. If the character is ever
1473  % made fully active, this will break our assumptions!
1474  %
1475  % If the char's meaning exists as a property list key, we
1476  % build up a chain of sub-/superscripts and iterate. (If not, exit and
1477  % typeset what we've already collected.)
1478  \cs_new:Nn \um_peek_execute_branches_ss_aux: {
1479    \prop_if_in:cxTF
1480      {g_um_\l_um_tmpa_tl _prop}
1481      {\meaning\l_peek_token}
1482      {
1483        \prop_get:cxN
1484          {g_um_\l_um_tmpa_tl _prop}
1485          {\meaning\l_peek_token}
1486          \l_um_tmpb_tl
1487        \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmpb_tl
1488        \l_peek_true_tl
1489      }
1490      {\l_peek_false_tl}
1491  }
```

8.0.4 Synonyms and all the rest

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

```
1492  \def\operator@font{\um_setup_mathup:}
```

```
1493  \def\to{\rightarrow}
1494  \def\vec{\overrightarrow}
1495  \def\le{\leq}
1496  \def\ge{\geq}
1497  \def\neq{\ne}
```

\mathcal

```
1498  \def\mathcal{\mathscr}
```

\mathrm

```
1499  \def\mathrm{\mathup}
```

8.0.5 Compatibility

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

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

```
1500    \@ifpackageloaded{amsmath}{
1501      \tl_remove_in:Nn \@begindocumenthook {
1502        \mathchardef\std@minus\mathcode`\-\relax
1503        \mathchardef\std@equal\mathcode`\=\relax
1504      }
1505    }{}
```

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

```
1506    \@ifpackageloaded{amsopn}{
1507      \cs_set:Npn \newmcodes@ {
1508        \mathcode`\'39
1509        \mathcode`\*42
1510        \mathcode`\."613A%
1511  %    \ifnum\mathcode`\-=45 \else
1512  %      \mathchardef\std@minus\mathcode`\-\relax
1513  %    \fi
1514        \mathcode`\-45
1515        \mathcode`\/47
1516        \mathcode`\:"603A\relax
1517      }
1518    }{}
```

Octothorpe is an odd one:

```
1519  \AtBeginDocument{
1520    \def\#{\mode_if_math:TF{\mathoctothorpe}{\char`\#}}
1521  }
```

Overriding amsmath definitions:

```
1522  \AtBeginDocument{
1523    \def\@cdots{\mathinner{\cdots}}
1524  }
```

Interaction with beamer:

```
1525  \@ifclassloaded{beamer}{
1526    \ifbeamer@suppressreplacements\else
1527      \PackageWarningNoLine{unicode-math}{
1528        Disabling~ beamer's~ math~ setup.^^J
1529        Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1530      }
1531      \beamer@suppressreplacementstrue
1532    \fi
1533  }{}
```

The end.

```
1534  \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.

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

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

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

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

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

Print the actual entry corresponding to the unicode character:

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

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

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

A 'fence' defined by the STIX table is something like \vert; in X{E}TEX this is just a \mathord that will grow with the magic of \XeTeXmathchardef.

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

Fixing up a couple of things in the STIX table.

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

A   Documenting maths support in the NFSS

A.1   Overview

In the following, ⟨*NFSS decl.*⟩ stands for something like {T1}{lmr}{m}{n}.

Maths symbol fonts Fonts for symbols: $\propto$, $\leq$, $\to$

> \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 un-affected 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 XƎTEX. 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 XƎTEX.

Accents are not included yet.

Summary   For symbols, something like:

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

For characters, something like:

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

File III

X﹁TEX math font dimensions

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

\fontdimen	Dimension name	Description
10	SCRIPTPERCENTSCALEDOWN	Percentage of scaling down for script level 1. Suggested value: 80%.
11	SCRIPTSCRIPTPERCENTSCALE-DOWN	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	DELIMITEDSUBFORMULAMIN-HEIGHT	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height × 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.

\fontdimen	Dimension name	Description
15	AxisHeight	Axis height of the font.
16	AccentBaseHeight	Maximum (ink) height of accent base that does not require raising the accents. Suggested: x-height of the font (os2.sxHeight) plus any possible overshots.
17	FlattenedAccentBaseHeight	Maximum (ink) height of accent base that does not require flattening the accents. Suggested: cap height of the font (os2.sCapHeight).
18	SubscriptShiftDown	The standard shift down applied to subscript elements. Positive for moving in the downward direction. Suggested: os2.ySubscriptYOffset.
19	SubscriptTopMax	Maximum allowed height of the (ink) top of subscripts that does not require moving subscripts further down. Suggested: /5 x-height.
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.

\fontdimen	Dimension name	Description
26	SuperscriptBottomMax-WithSubscript	The maximum level to which the (ink) bottom of superscript can be pushed to increase the gap between superscript and subscript, before subscript starts being moved down. Suggested: /5 x-height.
27	SpaceAfterScript	Extra white space to be added after each subscript and superscript. Suggested: 0.5pt for a 12 pt font.
28	UpperLimitGapMin	Minimum gap between the (ink) bottom of the upper limit, and the (ink) top of the base operator.
29	UpperLimitBaselineRiseMin	Minimum distance between baseline of upper limit and (ink) top of the base operator.
30	LowerLimitGapMin	Minimum gap between (ink) top of the lower limit, and (ink) bottom of the base operator.
31	LowerLimitBaselineDrop-Min	Minimum distance between baseline of the lower limit and (ink) bottom of the base operator.
32	StackTopShiftUp	Standard shift up applied to the top element of a stack.
33	StackTopDisplayStyleShift-Up	Standard shift up applied to the top element of a stack in display style.
34	StackBottomShiftDown	Standard shift down applied to the bottom element of a stack. Positive for moving in the downward direction.
35	StackBottomDisplayStyle-ShiftDown	Standard shift down applied to the bottom element of a stack in display style. Positive for moving in the downward direction.
36	StackGapMin	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element. Suggested: 3×default rule thickness.
37	StackDisplayStyleGapMin	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element in display style. Suggested: 7×default rule thickness.
38	StretchStackTopShiftUp	Standard shift up applied to the top element of the stretch stack.

\fontdimen	Dimension name	Description
39	STRETCHSTACKBOTTOMSHIFT-DOWN	Standard shift down applied to the bottom element of the stretch stack. Positive for moving in the downward direction.
40	STRETCHSTACKGAPABOVEMIN	Minimum gap between the ink of the stretched element, and the (ink) bottom of the element above. Suggested: UpperLimitGapMin
41	STRETCHSTACKGAPBELOWMIN	Minimum gap between the ink of the stretched element, and the (ink) top of the element below. Suggested: LowerLimitGapMin.
42	FRACTIONNUMERATORSHIFTUP	Standard shift up applied to the numerator.
43	FRACTIONNUMERATOR-DISPLAYSTYLESHIFTUP	Standard shift up applied to the numerator in display style. Suggested: StackTopDisplayStyleShiftUp.
44	FRACTIONDENOMINATORSHIFT-DOWN	Standard shift down applied to the denominator. Positive for moving in the downward direction.
45	FRACTIONDENOMINATOR-DISPLAYSTYLESHIFTDOWN	Standard shift down applied to the denominator in display style. Positive for moving in the downward direction. Suggested: StackBottomDisplayStyleShiftDown.
46	FRACTIONNUMERATORGAP-MIN	Minimum tolerated gap between the (ink) bottom of the numerator and the ink of the fraction bar. Suggested: default rule thickness
47	FRACTIONNUMDISPLAYSTYLE-GAPMIN	Minimum tolerated gap between the (ink) bottom of the numerator and the ink of the fraction bar in display style. Suggested: 3×default rule thickness.
48	FRACTIONRULETHICKNESS	Thickness of the fraction bar. Suggested: default rule thickness.
49	FRACTIONDENOMINATORGAP-MIN	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar. Suggested: default rule thickness

\fontdimen	Dimension name	Description
50	FractionDenomDisplayStyleGapMin	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar in display style. Suggested: 3×default rule thickness.
51	SkewedFractionHorizontalGap	Horizontal distance between the top and bottom elements of a skewed fraction.
52	SkewedFractionVerticalGap	Vertical distance between the ink of the top and bottom elements of a skewed fraction.
53	OverbarVerticalGap	Distance between the overbar and the (ink) top of he base. Suggested: 3×default rule thickness.
54	OverbarRuleThickness	Thickness of overbar. Suggested: default rule thickness.
55	OverbarExtraAscender	Extra white space reserved above the overbar. Suggested: default rule thickness.
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.

\fontdimen	Dimension name	Description
64	RADICALKERNAFTERDEGREE	Negative kern after the degree of a radical, if such is present. Suggested: −10/18 of em.
65	RADICALDEGREEBOTTOM-RAISEPERCENT	Height of the bottom of the radical degree, if such is present, in proportion to the ascender of the radical sign. Suggested: 60%.

File IV

Some manner of unit testing

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

B   The regular weight alphabets

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

```
1  (*test)
2  \documentclass{article}
3  \usepackage[a6paper]{geometry}
4  \usepackage{fontspec}
5  \setmainfont{TeX Gyre Pagella}
6  \usepackage{unicode-math}
7  \def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
8  \def\uplatin{abcdefghijklmnopqrstuvwxyz}
9  \def\upGreek{                            }
10 \def\upgreek{                              }
11 \def\itLatin{                         }
12 \def\itlatin{                          }
13 \def\itGreek{                      }
14 \def\itgreek{                           }
15 \def\testmath#1{%
16   \makebox[\linewidth][l]{%
17     \makebox[0pt][l]{$\csname up#1\endcsname$}%
18     \makebox[0pt][l]{$\csname it#1\endcsname$}}}
19 \begin{document}
20 \setmathfont[Colour=2255FF99]{Cambria Math}
21 \parindent=0pt
22 \voffset=-1in
```

```
23  \hoffset=-1in
24  \setbox0=\vbox{%
25  \testmath{Latin}\\
26  \testmath{latin}\\
27  \testmath{Greek}\\
28  \testmath{greek}}
29  \dimen0=\ht0
30  \advance\dimen0\dp0
31  \edef\papersize{papersize=\the\wd0,\the\dimen0}
32  \setbox255=\vbox{\special{\papersize}\box0}
33  \shipout\box255
34  \end{document}
35  ⟨/test⟩
```

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

C   The bold alphabets

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

```
36  ⟨*testbf⟩
37  \documentclass{article}
38  \usepackage[a6paper]{geometry}
39  \usepackage{fontspec}
40  \setmainfont{TeX Gyre Pagella}
41  \usepackage{unicode-math}
42  \def\upLatin{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
43  \def\uplatin{abcdefghijklmnopqrstuvwxyz}
44  \def\upGreek{                          }
45  \def\upgreek{                            }
46  \def\itLatin{                         }
47  \def\itlatin{                         }
48  \def\itGreek{                       }
49  \def\itgreek{                          }
50  \def\bfupLatin{                       }
51  \def\bfuplatin{                       }
52  \def\bfupGreek{                     }
53  \def\bfupgreek{                       }
54  \def\bfitLatin{                      }
55  \def\bfitlatin{                      }
56  \def\bfitGreek{                     }
57  \def\bfitgreek{                        }
58  \providecommand\mathalphabet{\mathbf}
```

78

```
59 \def\testmath#1{%
60   \makebox[\linewidth][l]{%
61     \makebox[0pt][l]{$\mathalphabet{\csname up#1\endcsname}$}%
62     \makebox[0pt][l]{$\mathalphabet{\csname it#1\endcsname}$}%
63     \makebox[0pt][l]{$\csname bfup#1\endcsname$}%
64     \makebox[0pt][l]{$\csname bfit#1\endcsname$}%
65     }}
66 \begin{document}
67 \setmathfont[Colour=2255FF55]{Cambria Math}
68 \parindent=0pt
69 \voffset=-1in
70 \hoffset=-1in
71 \setbox0=\vbox{%
72 \testmath{Latin}\\
73 \testmath{latin}\\
74 \testmath{Greek}\\
75 \testmath{greek}}
76 \dimen0=\ht0
77 \advance\dimen0\dp0
78 \edef\papersize{papersize=\the\wd0,\the\dimen0}
79 \setbox255=\vbox{\special{\papersize}\box0}
80 \shipout\box255
81 \end{document}
82 ⟨/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.

81

83