

Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/10/02 v0.4

Abstract

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

This package is intended to be a complete implementation of unicode maths for \LaTeX using the \XeTeX (and later, \LuaTeX) typesetting engines. With this package, changing maths fonts will be as easy as changing text fonts — not that there are many unicode maths fonts yet.

Maths input is simplified with unicode since literal glyphs may be entered instead of control sequences.

Contents

1	Introduction	3	7.3	The main <code>\setmathfont</code> macro	27
2	Acknowledgements	3	7.4	(Big) operators	35
3	Getting started	3	7.5	Radicals	39
3.1	Package options	3	7.6	Delimiters	39
4	Unicode maths font setup	4	7.7	Maths accents	41
4.1	Using multiple fonts	5	8	Font features	42
4.2	Script and scriptscript fonts/features	5	8.1	OpenType maths font features	43
5	Maths input	5	8.2	Script and scriptscript font options	43
5.1	Math ‘style’	6	8.3	Range processing	43
5.2	Bold style	7	8.4	Resolving Greek symbol name control sequences	47
5.3	Sans serif style	7	8.5	Setting up the mappings	49
5.4	All (the rest) of the mathematical alphabets	8	9	Maths alphabets mapping definitions	51
5.5	Miscellanea	10	9.1	Non-bold math alphabets	53
I	The unicode-math package	15	9.2	Bold math alphabets	57
6	Things we need	15	9.3	Definitions of the math symbols	64
6.1	Package options	19	10	Epilogue	65
6.2	Overcoming <code>\@on-</code> lypreamble	23	II	stix table data extraction	73
6.3	Other things	24	A	Documenting maths support in the NFSS	74
7	Fundamentals	25	A.1	Overview	74
7.1	Enlarging the number of maths families	25	III	X_YTeX math font dimensions	76
7.2	<code>\DeclareMathSymbol</code> for unicode ranges	25			

1 Introduction

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

Users who desire to specify maths alphabets only (Greek and Latin letters) may wish to use Andrew Moschou's mathspec package instead.

2 Acknowledgements

Many thanks to Microsoft for developing OpenType math as part of Office 2007; Jonathan Kew for implementing unicode math support in \XeTeX ; Barbara Beeton for her prodigious effort compiling the definitive list of unicode math glyphs and their \LaTeX names (inventing them where necessary), and also for her thoughtful replies to my sometimes incessant questions. Ross Moore and Chris Rowley have provided moral and technical support from the very early days with great insight into the issues we face trying to extend and use \TeX in the future. Apostolos Syropoulos, Joel Saloman, and Khaled Hosny have been fantastic beta testers.

3 Getting started

Load unicode-math as a regular \LaTeX package. It should be loaded after any other maths or font-related package in case it needs to overwrite their definitions. Here's an example:

```
\usepackage{amsmath} % if desired
\usepackage{unicode-math}
\setmathfont{Cambria Math}
```

3.1 Package options

Package options may be set when the package is loaded or at any later stage with the `\unimathsetup` command. Therefore, the following two examples are equivalent:

```
\usepackage[math-style=TeX]{unicode-math}
% OR
\usepackage{unicode-math}
\unimathsetup{math-style=TeX}
```

Table 1: Package options.

Option	Description	See...
<code>math-style</code>	Style of letters	section §5.1
<code>bold-style</code>	Style of bold letters	section §5.2
<code>sans-style</code>	Style of sans serif letters	section §5.3
<code>nabla</code>	Style of the nabla symbol	section §5.5.1
<code>partial</code>	Style of the partial symbol	section §5.5.2
<code>vargreek-shape</code>	Style of phi and epsilon	section §5.5.3
<code>colon</code>	Behaviour of <code>\colon</code>	section §5.5.6
<code>slash-delimiter</code>	Glyph to use for ‘stretchy’ slash	section §5.5.7

Note, however, that some package options affects how maths is initialised and changing an option such as `math-style` will not take effect until a new maths font is set up.

Package options may *also* be used when declaring new maths fonts, passed via options to the `\setmathfont` command. Therefore, the following two examples are equivalent:

```
\unimathsetup{math-style=TeX}
\setmathfont{Cambria Math}
% OR
\setmathfont[math-style=TeX]{Cambria Math}
```

A short list of package options is shown in table 1. See following sections for more information.

4 Unicode maths font setup

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

```
\setmathfont[<font features>]{<font name>}
```

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

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

Font features specific to `unicode-math` are shown in table 2. Package options (see table 1) may also be used. Other `fontspec` features are also valid.

Table 2: Maths font options.

Option	Description	See...
<code>range</code>	Style of letters	section §4.1
<code>ScriptFont</code>	Font to use for sub- and super-scripts	section §4.2
<code>ScriptFontFeatures</code>	Font features for sub- and super-scripts	section §4.2
<code>ScriptScriptFont</code>	Font to use for nested sub- and super-scripts	section §4.2
<code>ScriptScriptFontFeatures</code>	Font features for nested sub- and super-scripts	section §4.2

4.1 Using multiple fonts

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

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

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

4.2 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.

5 Maths input

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

Table 3: Effects of the `math-style` package option.

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

5.1 Math ‘style’

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

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

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

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

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

The `math-style` options’ effects are shown in brief in table 3.

Table 4: Effects of the `bold-style` package option.

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

5.2 Bold style

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

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

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

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

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

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

The `bold-style` options' effects are shown in brief in table 4.

5.3 Sans serif style

Unicode contains upright and italic, medium and bold mathematical alphabet characters. These may be explicitly selected with the `\mathsfup`, `\mathsfit`, `\mathbf{sfup}`, and `\mathbf{sfit}` commands discussed in section §5.4.

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

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

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

5.3.1 What about bold sans serif?

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

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

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

5.4 All (the rest) of the mathematical alphabets

Unicode contains separate codepoints for most if not all variations of alphabet shape one may wish to use in mathematical notation. The complete list is shown in table 5. 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.

5.4.1 Double-struck

The double-struck alphabet (also known as 'blackboard bold') consists of upright Latin letters `{a-z,A-Z}`, numerals `0-9`, summation symbol Σ , and four Greek letters only: `{\gamma,\Omega,\Gamma,\Pi}`.

While `\mathbb{\sum}` does produce a double-struck summation symbol, its

Table 5: 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. See main text for description of `\mathbbit`.

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

Table 6: The various forms of nabla.

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

limits aren't properly aligned (see section §??). Therefore, either the literal character or the control sequence `\Bbbsum` are recommended instead.

There are also five Latin *italic* double-struck letters: *$\mathbb{D}\mathbb{d}\mathbb{e}\mathbb{i}\mathbb{j}$* . These can be accessed (if not with their literal characters or control sequences) with the `\mathbb{it}` alphabet switch, but note that only those five letters will give the expected output.

5.5 Miscellanea

5.5.1 Nabla

The symbol ∇ comes in the six forms shown in table 6. We want an individual option to specify whether we want upright or italic nabla by default (when either upright or italic nabla is used in the source). \TeX classically uses an upright nabla, but iso standards differ (I think). The package options `nabla=upright` and `nabla=italic` switch between the two choices. This is then inherited through `\mathbf`; `\mathit` and `\mathup` can be used to force one way or the other.

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

5.5.2 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 7 for the variations on the partial differential symbol.

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

Table 7: 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	∂

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

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

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

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

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

5.5.4 Primes

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

If the font does not contain the pre-drawn glyphs or more than four primes are used, the single prime glyph is used multiple times with a negative kern to get the spacing right. There is no user interface to adjust this negative kern yet (be-

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

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

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

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

cause I haven’t decided what it should look like); if you need to, write something like this:

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

5.5.5 Unicode subscripts and superscripts

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

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

5.5.6 Colon

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

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

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

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

Table 8: Slashes and backslashes.

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

like a punctuation colon with additional space around it). Otherwise, `\colon` is made to output a colon with `\mathpunct` spacing.

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

5.5.7 Slashes and backslashes

There are several slash-like symbols defined in unicode. The complete list is shown in table 8.

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

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

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

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

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

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

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

²§4.4.5.11 [unicode://unicode.org/unicode/5.0.0/](#)

Presumably, U+29F5: REVERSE SOLIDUS OPERATOR is intended to be used in a similar way, but it could also (perhaps?) be used to represent ‘inverse division’: $\pi \approx 7 \setminus 22$.³ The L^AT_EX name for this character is `\setminus`.

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

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

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

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

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

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

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

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

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

5.5.8 Normalising some input characters

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

U+2511: LATIN SMALL LETTER ALPHA

U+25B: LATIN SMALL LETTER EPSILON

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

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

(Not yet implemented.)

File I

The unicode-math package

This is the package.

```
1 \ProvidesPackage{unicode-math}
2 [2009/10/02 v0.4 Unicode maths in XeLaTeX]
```

6 Things we need

Packages

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

Start using L^AT_EX3 — finally!

```
6 \ExplSyntaxOn
```

Counters and conditionals

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

For math-style:

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

For bold-style:

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

For nabla:

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

6.0.9 Alphabet unicode positions

Before we begin, let's define the positions of the various unicode alphabets so that our code is a little more readable.⁴

```
23 \def\um@usv@num{\0}
24 \def\um@usv@upLatin{\A}
25 \def\um@usv@uplatin{\a}
26 \def\um@usv@upGreek{"391}
27 \def\um@usv@upgreek{"3B1}
28 \def\um@usv@itLatin{"1D434}
29 \def\um@usv@itlatin{"1D44E}
30 \def\um@usv@itGreek{"1D6E2}
31 \def\um@usv@itgreek{"1D6FC}
32 \def\um@usv@bbnum{"1D7D8}
33 \def\um@usv@bbLatin{"1D538}
34 \def\um@usv@bblatin{"1D552}
35 \def\um@usv@scrLatin{"1D49C}
36 \def\um@usv@scrlatin{"1D4B6}
37 \def\um@usv@frakLatin{"1D504}
38 \def\um@usv@fraklatin{"1D51E}
39 \def\um@usv@sfnum{"1D7E2}
40 \def\um@usv@sfupnum{"1D7E2}
41 \def\um@usv@sfitnum{"1D7E2}
42 \def\um@usv@sfupLatin{"1D5A0}
43 \def\um@usv@sflatin {"1D5A0}
44 \def\um@usv@sfuplatin{"1D5BA}
45 \def\um@usv@sflatin{"1D5BA}
46 \def\um@usv@sfitLatin{"1D608}
47 \def\um@usv@sfitlatin{"1D622}
48 \def\um@usv@ttnum{"1D7F6}
49 \def\um@usv@ttLatin{"1D670}
50 \def\um@usv@ttlLatin{"1D68A}
```

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

Bold:

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

Greek variants:

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

Bold:

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

Italic Greek variants:

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

Bold italic:

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

Nabla:

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

Partial:

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

6.1 Package options

xkeyval's package support is used here. I'll switch over to l3keys2e at some stage.

`\unimathsetup` This macro can be used in lieu of or later to override options declared when the package is loaded.

```
132 \DeclareDocumentCommand \unimathsetup {m} {  
133   \setkeys{unicode-math.sty}{#1}  
134 }
```

math-style

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

```

170 \um@uppartialtrue
171 \um@bfupLatintrue
172 \um@bfuplatintrue
173 \um@upNabla true
174 \bool_set_true:N \g_um_upsans_bool
175 \bool_set_false:N \g_um_texgreek_bool
176 \or
177 \um@literaltrue
178 \um@bfliteraltrue
179 \bool_set_true:N \g_um_sfliteral_bool
180 \bool_set_false:N \g_um_texgreek_bool
181 \fi
182 }

```

bold-style

```

183 \define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,french,literal}{
184 \ifcase\@tempb\relax
185 \um@bfupGreekfalse
186 \um@bfupgreekfalse
187 \um@bfupLatinfalse
188 \um@bfuplatinfalse
189 \um@uppartialfalse
190 \or
191 \um@bfupGreektrue
192 \um@bfupgreekfalse
193 \um@bfupLatintrue
194 \um@bfuplatintrue
195 \um@uppartialfalse
196 \or
197 \um@bfupGreektrue
198 \um@bfupgreektrue
199 \um@bfupLatintrue
200 \um@bfuplatintrue
201 \um@uppartialtrue
202 \or
203 \um@bfliteraltrue
204 \fi
205 }
206 \cs_set:Nn \um_setup_bfshapes: {
207 \tl_set:Nx \um_bf_Greek_up_or_it_usv { \if@um@bfupGreek \um@usv@bfupGreek \else \um@usv@bfit
208 \tl_set:Nx \um_bf_greek_up_or_it_usv { \if@um@bfupgreek \um@usv@bfupgreek \else \um@usv@bfit
209 \tl_set:Nx \um_bf_Latin_up_or_it_usv { \if@um@bfupLatin \um@usv@bfupLatin \else \um@usv@bfit
210 \tl_set:Nx \um_bf_latin_up_or_it_usv { \if@um@bfuplatin \um@usv@bfuplatin \else \um@usv@bfit
211 }

```

sans-style

```

212 \bool_new:N \g_um_upsans_bool
213 \bool_new:N \g_um_sfliteral_bool
214 \define@choicekey*{unicode-math.sty}
215     {sans-style}[\@tempa\@tempb]{iso,tex,literal}{
216     \ifcase\@tempb\relax
217     \bool_set_false:N \g_um_upsans_bool
218     \or
219     \bool_set_true:N \g_um_upsans_bool
220     \or
221     \bool_set_true:N \g_um_sfliteral_bool
222     \fi
223 }
224 \cs_set:Nn \um_setup_sfshapes: {
225     \bool_if:NTF \g_um_upsans_bool {
226         \tl_set:Nn \um_sf_Latin_up_or_it_usv      { \um@usv@sflatin      }
227         \tl_set:Nn \um_sf_latin_up_or_it_usv      { \um@usv@sflatin      }
228         \tl_set:Nn \um_bfsf_Latin_up_or_it_usv     { \um@usv@bfsfupLatin  }
229         \tl_set:Nn \um_bfsf_latin_up_or_it_usv     { \um@usv@bfsfuplatin  }
230         \tl_set:Nn \um_bfsf_Greek_up_or_it_usv     { \um@usv@bfsfupGreek  }
231         \tl_set:Nn \um_bfsf_greek_up_or_it_usv     { \um@usv@bfsfupgreek  }
232     }{
233         \tl_set:Nn \um_sf_Latin_up_or_it_usv      { \um@usv@sfitLatin   }
234         \tl_set:Nn \um_sf_latin_up_or_it_usv      { \um@usv@sfitlatin   }
235         \tl_set:Nn \um_bfsf_Latin_up_or_it_usv     { \um@usv@bfsffitLatin }
236         \tl_set:Nn \um_bfsf_latin_up_or_it_usv     { \um@usv@bfsffitlatin }
237         \tl_set:Nn \um_bfsf_Greek_up_or_it_usv     { \um@usv@bfsfitGreek }
238         \tl_set:Nn \um_bfsf_greek_up_or_it_usv     { \um@usv@bfsfitgreek }
239     }
240 }

```

Symbol obliqueness

```

241 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
242     \ifcase\@tempb\relax
243     \@um@upNablatrue
244     \or
245     \@um@upNablafalse
246     \fi
247 }
248 \cs_set:Nn \um_setup_nabla: {
249     \if@um@upNabla
250         \tl_set:Nn \um_Nabla_up_or_it_usv      { \um@usv@Nabla }
251         \tl_set:Nn \um_bfNabla_up_or_it_usv     { \um@usv@bfNabla }
252         \tl_set:Nn \um_bfsfNabla_up_or_it_usv   { \um@usv@bfsfNabla }
253     \else
254         \tl_set:Nn \um_Nabla_up_or_it_usv      { \um@usv@itNabla }

```

```

255 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfitNabla }
256 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfitNabla }
257 \fi
258 }

259 \define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
260 \ifcase\@tempb\relax
261 \um@uppartialtrue
262 \or
263 \um@uppartialfalse
264 \fi
265 }

266 \cs_set:Nn \um_setup_partial: {
267 \ifum@uppartial
268 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@partial }
269 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfpartial }
270 \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfpartial }
271 \else
272 \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@itpartial }
273 \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfitpartial }
274 \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfitpartial }
275 \fi
276 }

```

Epsilon and phi shapes

```

277 \define@choicekey*{unicode-math.sty}{vargreek-shape}[\@tempa\@tempb]{unicode,TeX}{
278 \ifcase\@tempb\relax
279 \bool_set_false:N \g_um_texgreek_bool
280 \or
281 \bool_set_true:N \g_um_texgreek_bool
282 \fi
283 }

```

Colon style

```

284 \bool_new:N \g_um_literal_colon_bool
285 \define@choicekey*{unicode-math.sty}{colon}[\@tempa\@tempb]{literal,TeX}{
286 \ifcase\@tempb\relax
287 \bool_set_true:N \g_um_literal_colon_bool
288 \or
289 \bool_set_false:N \g_um_literal_colon_bool
290 \fi
291 }

```

Slash delimiter style

```

292 \define@choicekey*{unicode-math.sty}{slash-delimiter}[\@tempa\@tempb]{ascii,frac,div}{

```

```

293 \ifcase\@tempb\relax
294   \tl_set:Nn \g_um_slash_delimiter_usv {"002F}
295 \or
296   \tl_set:Nn \g_um_slash_delimiter_usv {"2044}
297 \or
298   \tl_set:Nn \g_um_slash_delimiter_usv {"2215}
299 \fi
300 }
301 \ExecuteOptionsX{math-style=TeX,slash-delimiter=ascii}
302 \ProcessOptionsX

```

6.2 Overcoming \@onlypreamble

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

```

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

```

```

334 \SetMathAlphabet
335 \SetMathAlphabet@
336 \DeclareMathAccent
337 \set@mathaccent
338 \DeclareMathSymbol
339 \set@mathchar
340 \set@mathsymbol
341 \DeclareMathDelimiter
342 \@xxDeclareMathDelimiter
343 \@DeclareMathDelimiter
344 \@xDeclareMathDelimiter
345 \set@mathdelimiter
346 \set@@mathdelimiter
347 \DeclareMathRadical
348 \mathchar@type
349 \DeclareSymbolFontAlphabet
350 \DeclareSymbolFontAlphabet@
351 }{
352 \tl_remove_in:Nn \@preamblecmds {\do#1}
353 }

```

6.3 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	<code>\font\tmpfont="Cambria Math"</code>
0.60	<code>\um@fontdimen@percent{10}{\tmpfont}\</code>
0.65	<code>\um@fontdimen@percent{11}{\tmpfont}\</code>
	<code>\um@fontdimen@percent{65}{\tmpfont}</code>

```

354 \def\um@fontdimen@percent#1#2{
355   0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax
356 }

```

`\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.

```

357 \def\um@scaled@apply#1#2#3{
358   \ifx#1\scriptstyle
359     #2\um@fontdimen@percent{10}\um@font#3
360   \else

```



```

361 \ifx#1\scriptscriptstyle
362 #2\um@fontdimen@percent{11}\um@font#3
363 \else
364 #2#3%
365 \fi
366 \fi
367 }

```

7 Fundamentals

7.1 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

```

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

```

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

7.2 `\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
370 \def \um@mathsymbol#1#2#3#4{
371 \expandafter\um@set@mathsymbol\csname sym#3\endcsname#1#2{#4}}

```

The final macros that actually define the maths symbol with X_ETeX 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.
372 \def\um@set@mathsymbol#1#2#3#4{

```

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

```
373 \ifx\mathop#3\relax
```

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

The active math char is defined to expand to the macro `\sumop`.

```
374 \begingroup
375   \char_make_active:n {#4}
376   \global\mathcode#4="8000\relax
377   \um@scanactivedef #4 \@nil { \csname\cs_to_str:N #2 op\endcsname }
378 \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`.

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

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

```
382   \cs_gset:cpn { \cs_to_str:N #2 op } {
383     \csname\string#2@sym\endcsname
384     \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
385     \ifin@
386       \expandafter\nolimits
387     \fi
388   }
```

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

```
389 \else
```

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

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

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

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

```

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

```

Accents

```

404     \ifx\mathaccent#3\relax
405     \cs_gset:Npx #2 {\XeTeXmathaccent "\mathchar@type#3 #1 #4\relax}
406   \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.

```

407     \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
408   \fi
409 \fi
410 \fi
411 \fi
412 \fi
413 }

```

`\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.

```

414 \cs_set:Nn \um_set_mathcode:nnnn {
415   \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
416 }

```

7.3 The main `\setmathfont` macro

Using a Range including large character sets such as `\mathrel`, `\mathalpha`, *etc.*, is *very slow*! I hope to improve the performance somehow.

`\setmathfont` [#1]: font features
 #2 : font name

```

417 \DeclareDocumentCommand \setmathfont { O{ } m } {

```

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

```

418     \let\glb@currsizel\relax

```

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

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

- Tell fontspec that maths font features are actually allowed.

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

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

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

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

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

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

```
425 \tl_set:Nn \l_um_script_features_tl {ScriptStyle}
426 \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
427 \tl_set:Nn \l_um_script_font_tl {#2}
428 \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.

```
429 \setkeys*{unicode-math.sty}{#1}
430 \cs_set:Npx \um_tmp: {
431   \exp_not:N \setkeys*[um]{options}{\exp_not:V \XKV@rm}
432 }
433 \um_tmp:
434 \edef\@tempa{\noexpand\zf@fontspec{
435   Script = Math,
436   SizeFeatures = {
437     {Size = \tf@size-} ,
438     {Size = \sf@size-\tf@size ,
439       Font = \l_um_script_font_tl ,
440       \l_um_script_features_tl
441     } ,
442     {Size = -\sf@size ,
443       Font = \l_um_sscript_font_tl ,
444       \l_um_sscript_features_tl
```

```

445     }
446   },
447   \XKV@rm
448   }{#2}
449 }
450 \@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`:

```

451 \font\um@font="#2"\relax
452 %% \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
453 %% \um@ot@math@true
454 %% \else
455 %% \PackageWarningNoLine{unicode-math}{
456 %%   The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
457 %%   Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
458 %%   in~ a~ substandard~ manner
459 %% }
460 %% \fi

```

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

- Math symbols are defined with `\UnicodeMathSymbol`; see section §7.3.1 for the individual definitions

```

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

```

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

```

476 \DeclareSymbolFont{\um_symfont_tl}
477   {\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`.

```
478 \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.)

```
479 \um_setup_shapes:
480 \um_remap_symbols:
481 \um_setup_mathactives:
482 \um_setup_delcodes:
483 \um_setup_alphanum:
484 \um_setup_alphabets:
```

End of the `\setmathfont` macro.

```
485 }

486 \cs_new:Nn \um_setup_shapes: {
487   \um_setup_nabla:
488   \um_setup_partial:
489   \um_setup_sfshapes:
490   \um_setup_bfshapes:
491 }
```

7.3.1 Functions for setting up symbols with mathcodes

`\um_process_symbol_noparse:nnnn` If the Range font feature has been used, then only a subset of the unicode glyphs are to be defined. See section §8.3 for the code that enables this.

```
492 \cs_set:Nn \um_process_symbol_noparse:nnnn {
493   \um@mathsymbol{#2}{#3}{\um_symfont_t1}{#1}
494 }

495 \cs_set:Nn \um_process_symbol_parse:nnnn {
496   \um@parse@term{#1}{#2}{#3}{
497     \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
498   }
499 }
```

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

```
\um_remap_symbol_noparse:nnn
\um_remap_symbol_parse:nnn
500 \cs_new:Nn \um_remap_symbols: {
```

```

501 \um_remap_symbol:nnn{\-}{\mathbin}{"02212}% hyphen to minus
502 \um_remap_symbol:nnn{\*}{\mathbin}{"02217}% text asterisk to "cen-
    tred asterisk"
503 \bool_if:NF \g_um_literal_colon_bool {
504   \um_remap_symbol:nnn{\:}{\mathrel}{"02236}% colon to ratio (i.e., punct to rel)
505 }
506 \if@um@literal
507   \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
508   \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
509   \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
510   \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
511 \else
512   \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um@usv@Nabla_up_or_it_usv}
513   \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um@usv@partial_up_or_it_usv}
514 \fi

```

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

```

515 \if@um@bfliteral
516   \um_remap_symbol:nnn {\um@usv@bfNabla}{\mathord}{\um@usv@bfNabla}
517   \um_remap_symbol:nnn {\um@usv@bfitNabla}{\mathord}{\um@usv@bfitNabla}
518   \um_remap_symbol:nnn {\um@usv@bfsfNabla}{\mathord}{\um@usv@bfsfNabla}
519   \um_remap_symbol:nnn {\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsfitNabla}
520   \um_remap_symbol:nnn {\um@usv@bfpartial}{\mathord}{\um@usv@bfpartial}
521   \um_remap_symbol:nnn {\um@usv@bfitpartial}{\mathord}{\um@usv@bfitpartial}
522   \um_remap_symbol:nnn {\um@usv@bfsfpartial}{\mathord}{\um@usv@bfsfpartial}
523   \um_remap_symbol:nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
524 \else
525   \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla}{\mathord}{\um@usv@bfNabla_up_or_it_usv}
526   \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsfNabla_up_or_it_usv}
527   \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um@usv@bfpartial_up_or_it_usv}
528   \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfpartial_up_or_it_usv}
529 \fi
530 }

```

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

```

531 \cs_new:Nn \um_remap_symbol_parse:nnn {
532   \um@parse@term {#3} {\@nil} {#2} {
533     \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
534   }
535 }
536 \cs_new:Nn \um_remap_symbol_noparse:nnn {
537   \clist_map_inline:nn {#1} {
538     \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
539   }
540 }

```

7.3.2 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:`

```
541 \cs_new:Nn \um_setup_mathactives: {  
542   \um_make_mathactive:nnn {"2032} \primesingle \mathord  
543 }
```

`\um_make_mathactive:nnn` : TODO : hook into range feature Makes #1 a mathactive char, and gives cs #2 the meaning of mathchar #1 with class #3. You are responsible for giving active #1 a particular meaning!

```
544 \cs_new:Nn \um_make_mathactive:nnn {  
545   \XeTeXmathchardef #2 = "\mathchar@type #3  
546                       \csname sym\um_symfont_tl\endcsname  
547                       #1 \scan_stop:  
548   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:  
549 }
```

7.3.3 Delimiter codes

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

`\um_setup_delcodes:`

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



```

569 \um_set_delcode:n {"21C5} % arrows up down
570 \um_set_delcode:n {"21F5} % arrows down up
571 \um_set_delcode:n {"21C8} % arrows up up
572 \um_set_delcode:n {"21CA} % arrows down down
573 \um_set_delcode:n {"21D1} % double up arrow
574 \um_set_delcode:n {"21D3} % double down arrow
575 \um_set_delcode:n {"21D5} % double updown arrow
576 \um_set_delcode:n {"21DE} % up arrow double stroke
577 \um_set_delcode:n {"21DF} % down arrow double stroke
578 \um_set_delcode:n {"21E1} % up arrow dashed
579 \um_set_delcode:n {"21E3} % down arrow dashed
580 }

```

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

```

581 \cs_new:Nn \um_set_delcode:nn {
582   \XeTeXdelcode#1 = \csname sym\um_symfont_tl\endcsname #2
583 }
584 \cs_new:Nn \um_set_delcode:n {
585   \XeTeXdelcode#1 = \csname sym\um_symfont_tl\endcsname #1
586 }

```

7.3.4 Maths alphabets' character mapping

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

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

```

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

```

Normal weight

```

590   \if@um@literal
591     \um_setup_literals:
592   \else
593     \um_setup_Latin:
594     \um_setup_latin:
595     \um_setup_Greek:
596     \um_setup_greek:
597   \fi

```

Bold

```

598   \if@um@bfliteral
599     \um_setup_bf_literals:

```

```

600 \else
601   \if@um@bfupLatin
602     \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfupLatin}
603   \else
604     \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
605   \fi
606   \if@um@bfuplatin
607     \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfuplatin}
608   \else
609     \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
610   \fi
611   \if@um@bfupGreek
612     \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfupGreek}
613     \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
614   \else
615     \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}
616     \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
617   \fi
618   \if@um@bfupgreek
619     \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
620     \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
621     \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
622     \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
623     \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
624     \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
625     \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
626   \else
627     \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
628     \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
629     \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
630     \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
631     \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
632     \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
633     \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
634   \fi
635 \fi
636 \else
: TODO : what is supposed to happen here?
637 \fi
638 }

```

7.3.5 Functions for setting up the maths alphabets

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

Adds `\um_set_mathcode:nnnn` declarations to the specified maths alphabet’s definition (e.g., `\um@mathscr`). Uses `\um@addto@mathmap` (below) to expand the name of the current symbol font.

```

639 \cs_set:Nn \um_mathmap_noparse:Nnn {
640   \clist_map_inline:nn {#2} {
641     \exp_args:No \um@addto@mathmap \um_symfont_tl {##1}{#1}{#3}
642   }
643 }

```

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

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

```

644 \cs_set:Nn \um_mathmap_parse:Nnn {
645   \clist_map_inline:Nn \um@char@num@range {
646     \ifnum##1=#3\relax
647       \clist_map_inline:nn {#2} {
648         \exp_args:No \um@addto@mathmap \um_symfont_tl {####1}{#1}{#3}
649       }
650     \fi
651   }
652 }

```

`\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.

```

653 \newcommand\um@addto@mathmap[4]{
654   \tl_put_right:cn {um_setup_\cs_to_str:N #3:} {
655     \um_set_mathcode:nnnn{#2}{\mathalpha}{#1}{#4}
656   }
657 }

```

7.4 (Big) operators

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











However, the limits aren’t set automatically; that is, we want to define, a la Plain \TeX etc., `\def\int{\intop\nolimits}`, so there needs to be a transforma-


tion from `\int` to `\intop` during the expansion of `\UnicodeMathSymbol` in the appropriate contexts.

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

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

U+029F8	$\big/$	<code>\xsol</code>	BIG SOLIDUS
U+029F9	$\big\backslash$	<code>\xbsol</code>	BIG REVERSE SOLIDUS
U+02A00	\bigodot	<code>\bigodot</code>	N-ARY CIRCLED DOT OPERATOR
U+02A01	\bigoplus	<code>\bigoplus</code>	N-ARY CIRCLED PLUS OPERATOR
U+02A02	\bigotimes	<code>\bigotimes</code>	N-ARY CIRCLED TIMES OPERATOR
U+02A03	$\bigcup\!\!\!\bigcup$	<code>\bigcupdot</code>	N-ARY UNION OPERATOR WITH DOT
U+02A04	$\bigcup\!\!\!\bigcup$	<code>\biguplus</code>	N-ARY UNION OPERATOR WITH PLUS
U+02A05	\bigsqcap	<code>\bigsqcap</code>	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06	\bigsqcup	<code>\bigsqcup</code>	N-ARY SQUARE UNION OPERATOR
U+02A07	\bigwedge	<code>\conjquant</code>	TWO LOGICAL AND OPERATOR
U+02A08	\bigvee	<code>\disjquant</code>	TWO LOGICAL OR OPERATOR
U+02A09	\bigtimes	<code>\bigtimes</code>	N-ARY TIMES OPERATOR
U+02A0B	\int_0^1	<code>\sumint</code>	SUMMATION WITH INTEGRAL
U+02A0C	\iiint_0^1	<code>\iiiint</code>	QUADRUPLE INTEGRAL OPERATOR
U+02A0D	\int_0^1	<code>\intbar</code>	FINITE PART INTEGRAL
U+02A0E	\int_0^1	<code>\intBar</code>	INTEGRAL WITH DOUBLE STROKE
U+02A0F	\int_0^1	<code>\fint</code>	INTEGRAL AVERAGE WITH SLASH
U+02A10	\oint_0^1	<code>\cirfnint</code>	CIRCULATION FUNCTION
U+02A11	\int_0^1	<code>\awint</code>	ANTICLOCKWISE INTEGRATION LINE INTEGRATION WITH RECTANGULAR
U+02A12	\int_0^1	<code>\rppolint</code>	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13	\int_0^1	<code>\scpolint</code>	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE
U+02A14	\int_0^1	<code>\npolint</code>	POLE
U+02A15	\oint_0^1	<code>\pointint</code>	INTEGRAL AROUND A POINT OPERATOR
U+02A16	\int_0^1	<code>\sqint</code>	QUATERNION INTEGRAL OPERATOR INTEGRAL WITH LEFTWARDS ARROW WITH
U+02A17	\int_0^1	<code>\intlarhk</code>	HOOK
U+02A18	\int_0^1	<code>\intx</code>	INTEGRAL WITH TIMES SIGN
U+02A19	\int_0^1	<code>\intcap</code>	INTEGRAL WITH INTERSECTION

U+02A1A		<code>\intcup</code>	INTEGRAL WITH UNION
U+02A1B		<code>\upint</code>	INTEGRAL WITH OVERBAR
U+02A1C		<code>\lowint</code>	INTEGRAL WITH UNDERBAR
U+02A1D		<code>\Join</code>	JOIN
U+02A1E		<code>\bigtriangleleft</code>	LARGE LEFT TRIANGLE OPERATOR
U+02A1F		<code>\zcmp</code>	Z NOTATION SCHEMA COMPOSITION
U+02A20		<code>\zpipe</code>	Z NOTATION SCHEMA PIPING
U+02A21		<code>\zproject</code>	Z NOTATION SCHEMA PROJECTION
U+02AFC		<code>\biginterleave</code>	LARGE TRIPLE VERTICAL BAR OPERATOR
U+02AFF		<code>\bigtalloblong</code>	N-ARY WHITE VERTICAL BAR

`\um@nolimits` This macro is a sequence containing those maths operators that require a `\nolimits` suffix. This list is used when processing `unicode-math-table.tex` to define such commands automatically (see the macro `\um@set@mathsymbol` on page 25). I’ve chosen essentially just the operators that look like integrals; hopefully a better mathematician can help me out here. I’ve a feeling that it’s more useful *not* to include the multiple integrals such as , but that might be a matter of preference.

```

658 \def\um@nolimits{
659   \@elt\int\@elt\iint\@elt\iiint\@elt\iiint\@elt\oint\@elt\oiint\@elt\oiint
660   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
661   \@elt\intbar\@elt\intBar\@elt\fint\@elt\cirfnint\@elt\awint\@elt\rppoint
662   \@elt\scpolint\@elt\ntopolint\@elt\pointint\@elt\sqint\@elt\intlarhk\@elt\intx
663   \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
664 }

```

`\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`.

```

665 \newcommand\addnolimits[1]{
666   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
667 }

```

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

```

668 \def\removenolimits#1{
669   \begingroup
670     \def\@elt##1{
671       \ifx##1#1\else

```

```

672     \noexpand\@elt\noexpand##1
673     \fi}
674     \xdef\um@nolimits{\um@nolimits}
675 \endgroup
676 }

```

7.5 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.

```

677 \def\um@radicals{\sqrt}

```

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

```

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

```

7.6 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>

```
























678 \let\left@primitive\left
679 \def\left{\mathopen{}\left@primitive}

```

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

Here are all `\mathopen` characters:

USV	Ex.	Macro	Description
U+00028	(<code>\lparen</code>	LEFT PARENTHESIS
U+0005B	[<code>\lbrack</code>	LEFT SQUARE BRACKET
U+0007B	{	<code>\lbrace</code>	LEFT CURLY BRACKET
U+0007C		<code>\lvert</code>	VERTICAL BAR
U+02016		<code>\lVert</code>	DOUBLE VERTICAL BAR
U+0221A	✓	<code>\sqrt</code>	RADICAL
U+0221B	✓ ³	<code>\cuberoot</code>	CUBE ROOT

U+0221C		<code>\fourthroot</code>	FOURTH ROOT
U+02308		<code>\lceil</code>	LEFT CEILING
U+0230A		<code>\lfloor</code>	LEFT FLOOR
U+0231C		<code>\ulcorner</code>	UPPER LEFT CORNER
U+0231E		<code>\llcorner</code>	LOWER LEFT CORNER
			LIGHT LEFT TORTOISE SHELL BRACKET
U+02772		<code>\lbrbrak</code>	ORNAMENT
U+027C5		<code>\lbag</code>	LEFT S-SHAPED BAG DELIMITER
U+027CC		<code>\longdivision</code>	LONG DIVISION
			MATHEMATICAL LEFT WHITE SQUARE
U+027E6		<code>\lBrack</code>	BRACKET
U+027E8		<code>\langle</code>	MATHEMATICAL LEFT ANGLE BRACKET
			MATHEMATICAL LEFT DOUBLE ANGLE
U+027EA		<code>\lAngle</code>	BRACKET
			MATHEMATICAL LEFT WHITE TORTOISE
U+027EC		<code>\Lbrbrak</code>	SHELL BRACKET
U+02983		<code>\lBrace</code>	LEFT WHITE CURLY BRACKET
U+02985		<code>\lParen</code>	LEFT WHITE PARENTHESIS
U+02987		<code>\llparenthesis</code>	Z NOTATION LEFT IMAGE BRACKET
U+02989		<code>\llangle</code>	Z NOTATION LEFT BINDING BRACKET
U+0298B		<code>\lbrackubar</code>	LEFT SQUARE BRACKET WITH UNDERBAR
			LEFT SQUARE BRACKET WITH TICK IN TOP
U+0298D		<code>\lbrackultick</code>	CORNER
			LEFT SQUARE BRACKET WITH TICK IN
U+0298F		<code>\lbracklltick</code>	BOTTOM CORNER
U+02991		<code>\langedot</code>	LEFT ANGLE BRACKET WITH DOT
U+02993		<code>\lparenless</code>	LEFT ARC LESS-THAN BRACKET
U+02997		<code>\lblkbrbrak</code>	LEFT BLACK TORTOISE SHELL BRACKET
U+029D8		<code>\lvzigzag</code>	LEFT WIGGLY FENCE
U+029DA		<code>\lvzigzag</code>	LEFT DOUBLE WIGGLY FENCE
U+029FC		<code>\lcurvyangle</code>	LEFT POINTING CURVED ANGLE BRACKET
U+03014		<code>\lbrbrak</code>	LEFT BROKEN BRACKET
U+03018		<code>\Lbrbrak</code>	LEFT WHITE TORTOISE SHELL BRACKET

And `\mathclose`:

USV	Ex.	Macro	Description
U+00029)	<code>\rparen</code>	RIGHT PARENTHESIS
U+0005D]	<code>\rbrack</code>	RIGHT SQUARE BRACKET
U+0007C		<code>\rvert</code>	VERTICAL BAR
U+0007D	}	<code>\rbrace</code>	RIGHT CURLY BRACKET
U+02016		<code>\rVert</code>	DOUBLE VERTICAL BAR
U+02309]	<code>\rceil</code>	RIGHT CEILING
U+0230B]	<code>\rfloor</code>	RIGHT FLOOR

U+0231D	↗	\urcorner	UPPER RIGHT CORNER
U+0231F	↘	\lrcorner	LOWER RIGHT CORNER
			LIGHT RIGHT TORTOISE SHELL BRACKET
U+02773		\rbrbrak	ORNAMENT
U+027C6	⌋	\rbag	RIGHT S-SHAPED BAG DELIMITER
			MATHEMATICAL RIGHT WHITE SQUARE
U+027E7	⌋	\rBrack	BRACKET
U+027E9	⌋	\rangle	MATHEMATICAL RIGHT ANGLE BRACKET
			MATHEMATICAL RIGHT DOUBLE ANGLE
U+027EB	⌋	\rAngle	BRACKET
			MATHEMATICAL RIGHT WHITE TORTOISE
U+027ED		\Rbrbrak	SHELL BRACKET
U+02984	⌋	\rBrace	RIGHT WHITE CURLY BRACKET
U+02986	⌋	\rParen	RIGHT WHITE PARENTHESIS
U+02988	⌋	\rrparenthesis	Z NOTATION RIGHT IMAGE BRACKET
U+0298A	⌋	\rrangle	Z NOTATION RIGHT BINDING BRACKET
U+0298C	⌋	\rbrackubar	RIGHT SQUARE BRACKET WITH UNDERBAR
			RIGHT SQUARE BRACKET WITH TICK IN
U+0298E	⌋	\rbracklrtick	BOTTOM CORNER
			RIGHT SQUARE BRACKET WITH TICK IN TOP
U+02990	⌋	\rbrackurtick	CORNER
U+02992	⌋	\rangledot	RIGHT ANGLE BRACKET WITH DOT
U+02994	⌋	\rparengtr	RIGHT ARC GREATER-THAN BRACKET
U+02998	⌋	\rblkrbrak	RIGHT BLACK TORTOISE SHELL BRACKET
U+029D9	⌋	\rvzigzag	RIGHT WIGGLY FENCE
U+029DB	⌋	\Rvzigzag	RIGHT DOUBLE WIGGLY FENCE
U+029FD	⌋	\rcurvyangle	RIGHT POINTING CURVED ANGLE BRACKET
U+03015		\rbrbrak	RIGHT BROKEN BRACKET
U+03019		\Rbrbrak	RIGHT WHITE TORTOISE SHELL BRACKET

7.7 Maths accents

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

USV	Ex.	Macro	Description
U+00300	˘	\grave	GRAVE ACCENT
U+00301	ˆ	\acute	ACUTE ACCENT
U+00302	ˆ	\hat	CIRCUMFLEX ACCENT
U+00303	˜	\tilde	TILDE
U+00304	ˉ	\bar	MACRON
U+00305	ˆ	\overbar	OVERBAR EMBELLISHMENT
U+00306	˘	\breve	BREVE
U+00307	˙	\dot	DOT ABOVE
U+00308	¨	\ddot	DIERESIS

U+00309	\acute{x}	<code>\ovhook</code>	COMBINING HOOK ABOVE
U+0030A	\mathring{x}	<code>\ocirc</code>	RING
U+0030C	\check{x}	<code>\check</code>	CARON
U+00310	\breve{x}	<code>\candra</code>	CANDRABINDU (NON-SPACING)
U+00312	\mathring{x}	<code>\oturnedcomma</code>	COMBINING TURNED COMMA ABOVE GREEK PSILI (SMOOTH BREATHING)
U+00313	\acute{x}	<code>\osmooth</code>	(NON-SPACING) GREEK DASIA (ROUGH BREATHING)
U+00314	\acute{x}	<code>\orough</code>	(NON-SPACING)
U+00315	\acute{x}	<code>\ocommatopright</code>	COMBINING COMMA ABOVE RIGHT
U+0031A	\acute{x}	<code>\droang</code>	LEFT ANGLE ABOVE (NON-SPACING) COMBINING LONG SOLIDUS
U+00338	$\not x$	<code>\not</code>	OVERLAY
U+020D0	\overleftarrow{x}	<code>\leftharpoonaccent</code>	COMBINING LEFT HARPOON ABOVE
U+020D1	\overrightarrow{x}	<code>\rightharpoonaccent</code>	COMBINING RIGHT HARPOON ABOVE
U+020D2	\overline{x}	<code>\vertoverlay</code>	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6	\overleftarrow{x}	<code>\overleftarrow</code>	COMBINING LEFT ARROW ABOVE
U+020D7	\overrightarrow{x}	<code>\overrightarrow</code>	COMBINING RIGHT ARROW ABOVE
U+020DB	\ddot{x}	<code>\dddot</code>	COMBINING THREE DOTS ABOVE
U+020DC	$\overset{\cdot\cdot\cdot}{x}$	<code>\ddddot</code>	COMBINING FOUR DOTS ABOVE
U+020E1	\overleftrightarrow{x}	<code>\overleftrightharpoon</code>	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7	\mathfrak{A}	<code>\annuity</code>	COMBINING ANNUITY SYMBOL
U+020E8	$\underset{\cdot\cdot\cdot}{x}$	<code>\threeunderdot</code>	COMBINING TRIPLE UNDERDOT
U+020E9	\overbrace{x}	<code>\widebridgeabove</code>	COMBINING WIDE BRIDGE ABOVE COMBINING RIGHTWARDS HARPOON WITH
U+020EC	\mathfrak{A}	<code>\underrightharpoondown</code>	BARB DOWNWARDS COMBINING LEFTWARDS HARPOON WITH
U+020ED	\mathfrak{A}	<code>\underleftharpoondown</code>	BARB DOWNWARDS
U+020EE	\mathfrak{A}	<code>\underleftarrow</code>	COMBINING LEFT ARROW BELOW
U+020EF	\mathfrak{A}	<code>\underrightarrow</code>	COMBINING RIGHT ARROW BELOW
U+020F0	\mathfrak{A}	<code>\asteraccent</code>	COMBINING ASTERISK ABOVE

8 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.

```

680 \newcommand\um@zf@feature[2]{
681   \define@key[zf]{options}{#1}[]{}
682   \if@um@fontspec@feature
683     #2
684   \else
685     \PackageError{fontspec/unicode-math}
```

```

686         {The '#1' font feature can only be used for maths fonts}
687         {The feature you tried to use can only be in commands
688         like \protect\setmathfont}
689     \fi
690 }
691 }

```

8.1 OpenType maths font features

```

692 \um@zf@feature{ScriptStyle}{
693     \zf@update@ff{+ssty=0}
694 }
695 \um@zf@feature{ScriptScriptStyle}{
696     \zf@update@ff{+ssty=1}
697 }

```

8.2 Script and scriptscript font options

```

698 \define@cmdkey[um]{options}[um@]{ScriptFeatures}{}
699 \define@cmdkey[um]{options}[um@]{ScriptScriptFeatures}{}
700 \define@cmdkey[um]{options}[um@]{ScriptFont}{}
701 \define@cmdkey[um]{options}[um@]{ScriptScriptFont}{}

```

8.3 Range processing

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

```

702 \define@choicekey+[um]{options}{Range}[\@tempa\@tempb]{ALL}{
703     \ifcase\@tempb\relax
704         \global\let\um@char@range\@empty
705     \fi
706 }{
707     \xdef\um@char@range{#1}
708 }

```

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 13.

Table 13: 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:

```

709 \newcommand\um@parse@term[4]{
710   \clist_map_variable:NNn \um@char@range \@ii {
711     \unless\ifx\@ii\@empty
712       \@tempswafalse

```

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

```

713     \expandafter\um@firstchar\expandafter{\@ii}
714     \ifx\@tempa\um@backslash
715       \expandafter\ifx\@ii#2\relax
716         \@tempswatrue
717     \else
718       \expandafter\ifx\@ii#3\relax
719         \@tempswatrue
720     \fi
721   \fi

```

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

```

722   \else
723     \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
724   \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.

```

725   \if@tempswa
726     \ifx\um@char@num@range\@empty
727       \g@addto@macro\um@char@num@range{#1}
728     \else
729       \g@addto@macro\um@char@num@range{,#1}
730     \fi
731     #4%
732   \fi
733 \fi
734 }
735 }
736 \def\um@firstof#1#2\@nil{#1}
737 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}

```

738 \def\um@firstchar#1{\edef\@tempa{\expandafter\um@firstof\string#1\@nil}}

<p>'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</p>	<pre> \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} </pre>
--	---

\um@parse@range Weird syntax. As shown previously in table 13, this macro can be passed four different input types via \um@parse@term.

```

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

```

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

```

742   \expandafter\ifx\expandafter\@marker\@tempb\relax
743     \ifnum#4=#1\relax
744       \@tempwattrue
745     \fi
746   \else

```

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

```

747   \ifx\@empty\@tempb
748     \ifnum#4>\numexpr#1-1\relax
749       \@tempwattrue
750     \fi
751   \else

```

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

```

752   \ifx\@empty\@tempa
753     \ifnum#4<\numexpr#2+1\relax
754       \@tempwattrue
755     \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-

```

756     \else
757         \ifnum#4>\numexpr#1-1\relax
758         \ifnum#4<\numexpr#2+1\relax
759             \@tempwattrue
760         \fi
761     \fi
762 \fi
763 \fi
764 \fi
765 }

```

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

Loops through character ranges setting \mathcode.

```

766 \cs_set:Nn \um_map_chars_range:nnn {
767     \clist_map_variable:nnN {#2} \l_um_input_num {
768         \prg_stepwise_variable:nnnNn{0}{1}{#1} \l_um_incr_num {
769             \um_set_mathcode:nnnn
770             {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
771             {\mathalpha}{\um_symfont_t1}
772             {\numexpr \l_um_incr_num + #3 \relax}
773         }
774     }
775 }
776 \cs_set:Nn \um_map_chars_latin:nn {
777     \um_map_chars_range:nnn {25}{#1}{#2}
778 }
779 \cs_set:Nn \um_map_chars_greek:nn {
780     \um_map_chars_range:nnn {24}{#1}{#2}
781 }
782 \cs_set:Nn \um_map_chars_numbers:nn {
783     \um_map_chars_range:nnn {9}{#1}{#2}
784 }
785 \cs_set:Nn \um_map_char:nn {
786     \um_map_chars_range:nnn {0}{#1}{#2}
787 }

```

\um_set_mathalphabet_char:Nnn #1 : Maths alphabet
#2 : Input char(s)
#3 : Output char

Loops through character ranges setting \mathcode.

```

788 \cs_set:Npn \exp_args:Nnff {\::n\::f\::f\:::}
789 \cs_new:Nn \um_set_mathalphabet_char:Nnn {
790   \clist_map_variable:nNn {#2} \l_um_input_num {
791     \exp_args:Nnff \um_mathmap:Nnn {#1}
792     {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
793   }
794 }

```

`\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`.

```

795 \cs_new:Nn \um_set_mathalph_range:nNnn {
796   \clist_map_variable:nNn {#3} \l_um_input_num {
797     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
798       \exp_args:Nnff \um_mathmap:Nnn {#2}
799       {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
800       {\number\numexpr \l_um_inc_num + #4 \relax}
801     }
802   }
803 }
804 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
805   \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
806 }
807 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
808   \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
809 }
810 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
811   \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
812 }

```

BCDBCDEABCDEF

```

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

```

8.4 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.

```

813 \AtBeginDocument{\um@resolve@greek}
814 \newcommand\um@resolve@greek{
815   \def\Alpha{\mitAlpha}

```

```

816 \def\Beta{\mitBeta}
817 \def\Gamma{\mitGamma}
818 \def\Delta{\mitDelta}
819 \def\Epsilon{\mitEpsilon}
820 \def\Zeta{\mitZeta}
821 \def\Eta{\mitEta}
822 \def\Theta{\mitTheta}
823 \def\Iota{\mitIota}
824 \def\Kappa{\mitKappa}
825 \def\Lambda{\mitLambda}
826 \def\Mu{\mitMu}
827 \def\Nu{\mitNu}
828 \def\Xi{\mitXi}
829 \def\Omicron{\mitOmicron}
830 \def\Pi{\mitPi}
831 \def\Rho{\mitRho}
832 \def\varTheta{\mitvarTheta}
833 \def\Sigma{\mitSigma}
834 \def\Tau{\mitTau}
835 \def\Upsilon{\mitUpsilon}
836 \def\Phi{\mitPhi}
837 \def\Chi{\mitChi}
838 \def\Psi{\mitPsi}
839 \def\Omega{\mitOmega}

```

Lowercase:

```

840 \def\alpha{\mitalpha}
841 \def\beta{\mitbeta}
842 \def\gamma{\mitgamma}
843 \def\delta{\mitdelta}
844 \def\epsilon{
845   \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
846 }
847 \def\zeta{\mitzeta}
848 \def\eta{\miteta}
849 \def\theta{\mittheta}
850 \def\iota{\mitiota}
851 \def\kappa{\mitkappa}
852 \def\lambda{\mitlambda}
853 \def\mu{\mitmu}
854 \def\nu{\mitnu}
855 \def\xi{\mitxi}
856 \def\omicron{\mitomicron}
857 \def\pi{\mitpi}
858 \def\rho{\mitrho}
859 \def\varsigma{\mitvarsigma}
860 \def\sigma{\mitsigma}

```



```

861 \def\tau{\mittau}
862 \def\upsilon{\mitupsilon}
863 \def\phi{
864   \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}
865 }
866 \def\chi{\mitchi}
867 \def\psi{\mitpsi}
868 \def\omega{\mitomega}
869 \def\varepsilon{
870   \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
871 }
872 \def\vartheta{\mitvartheta}
873 \def\varkappa{\mitvarkappa}
874 \def\varphi{
875   \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
876 }
877 \def\varrho{\mitvarrho}
878 \def\varpi{\mitvarpi}
879 }

```

8.5 Setting up the mappings

\um_setup_literals: : TODO : other literal symbols

```

880 \cs_set:Nn \um_setup_literals: {
881   \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}
882   \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}
883   \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}
884   \um_map_char:nn {\um@usv@ith}{\um@usv@ith}
885   \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}
886   \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}
887   \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}
888   \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}
889   \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}
890 }

```

\um_setup_bf_literals: TODO: other literal symbols

```

891 \cs_set:Nn \um_setup_bf_literals: {
892   \um_map_chars_latin:nn {\um@usv@bfupLatin}{\um@usv@bfupLatin}
893   \um_map_chars_latin:nn {\um@usv@bfuplatin}{\um@usv@bfuplatin}
894   \um_map_chars_latin:nn {\um@usv@bfitLatin}{\um@usv@bfitLatin}
895   \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}
896   \um_map_chars_greek:nn {\um@usv@bfupGreek}{\um@usv@bfupGreek}
897   \um_map_chars_greek:nn {\um@usv@bfupgreek}{\um@usv@bfupgreek}
898   \um_map_chars_greek:nn {\um@usv@bfitGreek}{\um@usv@bfitGreek}
899   \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}
900 }

```

`\um_setup_Latin:`

```
901 \cs_set:Nn \um_setup_Latin: {  
902   \if@um@upLatin  
903     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}  
904   \else  
905     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}  
906   \fi  
907 }
```

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

```
908 \cs_set:Nn \um_setup_latin: {  
909   \if@um@uplatin  
910     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}  
911     \um_map_char:nn {\um@usv@ith}{`\h}  
912   \else  
913     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}  
914     \um_map_char:nn {\h,\um@usv@ith}{\um@usv@ith}  
915   \fi  
916 }
```

`\um_setup_Greek:`

```
917 \cs_set:Nn \um_setup_Greek: {  
918   \if@um@upGreek  
919     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}  
920     \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}  
921   \else  
922     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}  
923     \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}  
924   \fi  
925 }
```

`\um_setup_greek:`

```
926 \cs_set:Nn \um_setup_greek: {  
927   \if@um@upgreek  
928     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}  
929     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}  
930     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}  
931     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}  
932     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}  
933     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}  
934     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}  
935   \else  
936     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}  
937     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}  
938     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
```

```

939 \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
940 \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
941 \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
942 \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
943 \fi
944 }

```

9 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 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.

```

945 \cs_new:Nn \um_setup_alphabets: {
946   \um_setup_math_alphabet:nn {up}    {}{latin, Latin, greek, Greek}
947   \um_setup_math_alphabet:nn {it}    {}{latin, Latin, greek, Greek}
948   \um_setup_math_alphabet:nn {bb}    {}{latin, Latin, num}
949   \um_setup_math_mapping:n {bb}     {}
950   \um_maybe_init_alphabet:n {bbit}   {}
951   \um_setup_math_mapping:n {bbit}   {}
952   \um_setup_math_alphabet:nn {scr}   {}{latin, Latin}
953   \um_setup_math_alphabet:nn {frak}  {}{latin, Latin}
954   \um_setup_math_alphabet:nn {sf}    {}{latin, Latin, num}
955   \um_setup_math_alphabet:nn {sfup}  {}{latin, Latin, num}
956   \um_setup_math_alphabet:nn {sfit}  {}{latin, Latin, num}

```

```

957 \um_setup_math_alphabet:nn {tt }{{latin, Latin, num}}
958 \um_setup_math_alphabet:nn {bf }{{latin, Latin, greek, Greek, num}}
959 \um_setup_math_alphabet:nn {bfup }{{latin, Latin, greek, Greek, num}}
960 \um_setup_math_alphabet:nn {bfit }{{latin, Latin, greek, Greek, num}}
961 \um_setup_math_alphabet:nn {bfscr }{{latin, Latin}}
962 \um_setup_math_alphabet:nn {bffrak}{{latin, Latin}}
963 \um_setup_math_alphabet:nn {bfsf }{{latin, Latin, greek, Greek, num}}
964 \um_setup_math_alphabet:nn {bfsfup}{{latin, Latin, greek, Greek, num}}
965 \um_setup_math_alphabet:nn {bfsfit}{{latin, Latin, greek, Greek, num}}
966 }

```

\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.

```

967 \cs_new:Nn \um_setup_math_alphabet:nn {
968   \clist_map_inline:nn {#2} {
969     \um_glyph_if_exist:nT {\csname um@usv@#1##1 \endcsname}{
970       \um_maybe_init_alphabet:n {#1}
971       \clist_map_break:
972     }
973   }
974   \clist_map_inline:nn {#2} {
975     \um_glyph_if_exist:nTF {\csname um@usv@#1##1 \endcsname}{
976       \use:c {um_config_math#1_#1:}
977     }{
978       \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
979       Math~ alphabet~
980       \@backslashchar math#1~
981       (\tl_use:c{g_um_math_alphabet_name_#1_tl})~
982       not~ found~ in~ font~
983       \fontname\um@font}
984     }
985   }
986 }
987 \tl_set:Nn \g_um_math_alphabet_name_latin_tl {Latin, lowercase}
988 \tl_set:Nn \g_um_math_alphabet_name_Latin_tl {Latin, uppercase}
989 \tl_set:Nn \g_um_math_alphabet_name_greek_tl {Greek, lowercase}
990 \tl_set:Nn \g_um_math_alphabet_name_Greek_tl {Greek, uppercase}
991 \tl_set:Nn \g_um_math_alphabet_name_num_tl {Numerals}
992 \cs_new:Nn \um_setup_math_mapping:n {
993   \cs_if_exist:cT {um_setup_math#1:} {
994     \use:c {um_config_math#1_misc:}
995   }
996 }
997 \cs_set:Nn \um_init_alphabet:n {

```

```

998 \um_prepare_alph:n {#1}
999 \cs_set_eq:CN {um_setup_math#1:} \prg_do_nothing:
1000 }

```

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

```

1001 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
1002 \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
1003 }

```

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

```

1004 \cs_new:Nn \um_prepare_alph:n {
1005 \cs_if_exist:cF {um_math#1:n} {
1006 \cs_set:cpn {um_math#1:n} ##1 {
1007 \use:c {um_setup_math#1:} ##1 \egroup
1008 }
1009 \cs_set_protected:cpn {math#1} {
1010 \bgroup
1011 \mode_if_math:F {
1012 \egroup\expandafter
1013 \non@alpherr\expandafter{\csname math#1\endcsname\space}
1014 }
1015 \use:c {um_math#1:n}
1016 }
1017 }
1018 }

```

: TODO : nested alphabets?

9.1 Non-bold math alphabets

9.1.1 Upright: `\mathup`

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

```

1019 \cs_new:Npn \um_config_mathup_Latin: {
1020 \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
1021 }
1022 \cs_new:Npn \um_config_mathup_latin: {
1023 \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
1024 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@ith} {\` \h}
1025 }
1026 \cs_new:Npn \um_config_mathup_Greek: {
1027 \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
1028 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}
1029 \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varThe

```

```

1030 }
1031 \cs_new:Npn \um_config_mathup_greek: {
1032   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
1033   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}
1034   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@va
1035   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@varthe
1036   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkap
1037   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
1038   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
1039   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
1040 }

```

9.1.2 Italic: `\mathit`

```

1041 \cs_new:Npn \um_config_mathit_Latin: {
1042   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
1043 }
1044 \cs_new:Npn \um_config_mathit_latin: {
1045   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
1046   \um_set_mathalphabet_char:Nnn{\mathit}{\`h,\um@usv@ith}{\um@usv@ith}
1047 }
1048 \cs_new:Npn \um_config_mathit_Greek: {
1049   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
1050   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarT
1051   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
1052 }
1053 \cs_new:Npn \um_config_mathit_greek: {
1054   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
1055   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartia
1056   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@it
1057   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvar
1058   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvark
1059   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
1060   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
1061   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
1062 }

```

9.1.3 Blackboard or double-struck: `\mathbb` and `\mathbbi`

: TODO : make bbit work with literal input?

```

1063 \cs_new:Npn \um_config_mathbb_latin: {
1064   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}
1065   \um_set_mathalphabet_char:Nnn{\mathbb}{\um@usv@ith} {"1D559}
1066 }
1067 \cs_new:Npn \um_config_mathbb_Latin: {
1068   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
1069   \um_set_mathalphabet_char:Nnn{\mathbb}{\`C,"1D436}{ "2102}

```

```

1070 \um_set_mathalphabet_char:Nnn{\mathbb}{`\H,"1D43B}{ "210D}
1071 \um_set_mathalphabet_char:Nnn{\mathbb}{`\N,"1D441}{ "2115}
1072 \um_set_mathalphabet_char:Nnn{\mathbb}{`\P,"1D443}{ "2119}
1073 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Q,"1D444}{ "211A}
1074 \um_set_mathalphabet_char:Nnn{\mathbb}{`\R,"1D445}{ "211D}
1075 \um_set_mathalphabet_char:Nnn{\mathbb}{`\Z,"1D44D} { "2124}
1076 }
1077 \cs_new:Npn \um_config_mathbb_num: {
1078   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@num}{\um@usv@bnum}
1079 }
1080 \cs_new:Npn \um_config_mathbb_misc: {
1081   \um_set_mathalphabet_char:Nnn \mathbb {"03A0,"1D6F1}{ "213F} % Pi
1082   \um_set_mathalphabet_char:Nnn \mathbb {"03C0,"1D70B}{ "213C} % pi
1083   \um_set_mathalphabet_char:Nnn \mathbb {"0393,"1D6E4}{ "213E} % Gamma
1084   \um_set_mathalphabet_char:Nnn \mathbb {"03B3,"1D6FE}{ "213D} % gamma
1085   \um_set_mathalphabet_char:Nnn \mathbb {"2211}{ "2140} % summation
1086 }
1087 \cs_new:Npn \um_config_mathbbbit_misc: {
1088   \um_set_mathalphabet_char:Nnn \mathbbbit {\D,"1D437}{ "2145}
1089   \um_set_mathalphabet_char:Nnn \mathbbbit {\d,"1D451}{ "2146}
1090   \um_set_mathalphabet_char:Nnn \mathbbbit {\e,"1D452}{ "2147}
1091   \um_set_mathalphabet_char:Nnn \mathbbbit {\i,"1D456}{ "2148}
1092   \um_set_mathalphabet_char:Nnn \mathbbbit {\j,"1D457}{ "2149}
1093 }

```

9.1.4 Script or caligraphic: \mathscr and \mathcal

```

1094 \cs_new:Npn \um_config_mathscr_Latin: {
1095   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}
1096   \um_set_mathalphabet_char:Nnn \mathscr {\B,"1D435}{ "212C}
1097   \um_set_mathalphabet_char:Nnn \mathscr {\E,"1D438}{ "2130}
1098   \um_set_mathalphabet_char:Nnn \mathscr {\F,"1D439}{ "2131}
1099   \um_set_mathalphabet_char:Nnn \mathscr {\H,"1D43B}{ "210B}
1100   \um_set_mathalphabet_char:Nnn \mathscr {\I,"1D43C}{ "2110}
1101   \um_set_mathalphabet_char:Nnn \mathscr {\L,"1D43F}{ "2112}
1102   \um_set_mathalphabet_char:Nnn \mathscr {\M,"1D440}{ "2133}
1103   \um_set_mathalphabet_char:Nnn \mathscr {\R,"1D445}{ "211B}
1104 }
1105 \cs_new:Npn \um_config_mathscr_latin: {
1106   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrlatin}
1107   \um_set_mathalphabet_char:Nnn \mathscr {\e,"1D452}{ "212F}
1108   \um_set_mathalphabet_char:Nnn \mathscr {\g,"1D454}{ "210A}
1109   \um_set_mathalphabet_char:Nnn \mathscr {\o,"1D45C}{ "2134}
1110   \um_set_mathalphabet_char:Nnn \mathscr {\um@usv@ith} { "1D4BD}
1111 }

```

9.1.5 Fraktur or fraktur or blackletter: `\mathfrak`

```
1112 \cs_new:Npn \um_config_mathfrak_Latin: {  
1113   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLat  
1114   \um_set_mathalphabet_char:Nnn \mathfrak {\`C,"1D436}{ "212D}  
1115   \um_set_mathalphabet_char:Nnn \mathfrak {\`H,"1D43B}{ "210C}  
1116   \um_set_mathalphabet_char:Nnn \mathfrak {\`I,"1D43C}{ "2111}  
1117   \um_set_mathalphabet_char:Nnn \mathfrak {\`R,"1D445}{ "211C}  
1118   \um_set_mathalphabet_char:Nnn \mathfrak {\`Z,"1D44D}{ "2128}  
1119 }  
1120 \cs_new:Npn \um_config_mathfrak_latin: {  
1121   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklati  
1122   \um_set_mathalphabet_char:Nnn \mathfrak {\um@usv@ith} {"1D525}  
1123 }
```

9.1.6 Sans serif: `\mathsf`

```
1124 \cs_new:Npn \um_config_mathsf_Latin: {  
1125   \bool_if:NTF \g_um_sfliteral_bool {  
1126     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin}{\um@usv@sfupLatin}  
1127     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itLatin}{\um@usv@sfitLatin}  
1128   }{  
1129     \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@upLatin,\um@usv@itLatin}{ \um_sf_Latin_up  
1130   }  
1131 }  
1132 \cs_new:Npn \um_config_mathsf_latin: {  
1133   \bool_if:NTF \g_um_sfliteral_bool {  
1134     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin}{\um@usv@sfuplatin}  
1135     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itlatin}{\um@usv@sfitlatin}  
1136     \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}  
1137   }{  
1138     \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@uplatin,\um@usv@itlatin}{ \um_sf_latin_up  
1139     \bool_if:NTF \g_um_upsans_bool {  
1140       \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D5C1}  
1141     }{  
1142       \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}  
1143     }  
1144   }  
1145 }  
1146 \cs_new:Npn \um_config_mathsf_num: {  
1147   \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnun}  
1148 }
```

9.1.7 Sans serif upright: `\mathsfup`

```
1149 \cs_new:Npn \um_config_mathsfup_num: {  
1150   \um_set_mathalphabet_numbers:Nnn{\mathsfup}{\um@usv@num}{\um@usv@sfnun}  
1151 }  
1152 \cs_new:Npn \um_config_mathsfup_latin: {
```



```

1153 \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfuplat
1154 \um_set_mathalphabet_char:Nnn \mathsfup {\um@usv@ith} {"1D5C1}
1155 }
1156 \cs_new:Npn \um_config_mathsfup_Latin: {
1157 \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfupLat
1158 }

```

9.1.8 Sans serif italic: `\mathsf`

Map the numbers like that because it seems sensible.

```

1159 \cs_new:Npn \um_config_mathsf_num: {
1160 \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnum}
1161 }
1162 \cs_new:Npn \um_config_mathsf_Latin: {
1163 \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLat
1164 }
1165 \cs_new:Npn \um_config_mathsf_latin: {
1166 \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlat
1167 \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}
1168 }

```

9.1.9 Typewriter or monospaced: `\mathtt`

```

1169 \cs_new:Npn \um_config_mathtt_num: {
1170 \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
1171 }
1172 \cs_new:Npn \um_config_mathtt_Latin: {
1173 \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
1174 }
1175 \cs_new:Npn \um_config_mathtt_latin: {
1176 \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlatin}
1177 \um_set_mathalphabet_char:Nnn \mathtt {\um@usv@ith} {"1D691}
1178 }

```

9.2 Bold math alphabets

9.2.1 Bold: `\mathbf`

```

1179 \cs_new:Npn \um_config_mathbf_num: {
1180 \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
1181 }
1182 \cs_new:Npn \um_config_mathbf_Latin: {
1183 \if@um@bfliteral
1184 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfupLatin}
1185 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
1186 \else
1187 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bf_Latin_up_
1188 \fi

```

```

1189 }
1190 \cs_new:Npn \um_config_mathbf_latin: {
1191   \if@um@bfliteral
1192     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@bfuplatin}
1193     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@bfitlatin}
1194     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1195   \else
1196     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um_bf_latin_up_
1197     \if@um@bfuplatin
1198       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
1199     \else
1200       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1201     \fi
1202   \fi
1203 }
1204 \cs_new:Npn \um_config_mathbf_Greek: {
1205   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
1206   \if@um@bfliteral
1207     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfupGreek}
1208     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
1209     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
1210     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
1211     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
1212     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
1213   \else
1214     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um_bf_Greek_up_
1215     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um_bfNabla_up_or_i
1216     \if@um@bfupGreek
1217       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfv
1218     \else
1219       \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfit
1220     \fi
1221   \fi
1222 }
1223 \cs_new:Npn \um_config_mathbf_greek: {
1224   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
1225   \if@um@bfliteral
1226     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfupgreek}
1227     \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitgreek}
1228     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
1229     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bfvarepsilon}
1230     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
1231     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
1232     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
1233     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
1234     \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}

```

```

1235 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
1236 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
1237 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
1238 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1239 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
1240 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
1241 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
1242 \else
1243 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}
1244 \if@um@bfupgreek
1245 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
1246 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitvartheta}
1247 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1248 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi}
1249 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho}
1250 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1251 \else
1252 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}
1253 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfitvartheta}
1254 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1255 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarphi}
1256 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarrho}
1257 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1258 \fi
1259 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpartial}
1260 \fi
1261 }

```

9.2.2 Bold Italic: `\mathbfit`

```

1262 \cs_new:Npn \um_config_mathbfit_num: {
1263   \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnum}
1264 }
1265 \cs_new:Npn \um_config_mathbfit_Latin: {
1266   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1267 }
1268 \cs_new:Npn \um_config_mathbfit_latin: {
1269   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
1270   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@ith} {"1D489}
1271 }
1272 \cs_new:Npn \um_config_mathbfit_Greek: {
1273   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
1274   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
1275   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}
1276 }
1277 \cs_new:Npn \um_config_mathbfit_greek: {
1278   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}

```

```

1279 \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgre
1280 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpa
1281 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
1282 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfit
1283 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfit
1284 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarp
1285 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarr
1286 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1287 }

```

9.2.3 Bold Italic: `\mathbfup`

```

1288 \cs_new:Npn \um_config_mathbfup_num: {
1289   \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnum}
1290 }
1291 \cs_new:Npn \um_config_mathbfup_Latin: {
1292   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLat
1293 }
1294 \cs_new:Npn \um_config_mathbfup_latin: {
1295   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplat
1296   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@ith} {"1D421}
1297 }
1298 \cs_new:Npn \um_config_mathbfup_Greek: {
1299   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGre
1300   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfva
1301   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}
1302   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Digamma}{\um@usv@bfDigamma}
1303 }
1304 \cs_new:Npn \um_config_mathbfup_greek: {
1305   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgre
1306   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpart
1307   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
1308   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfva
1309   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfva
1310   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi
1311   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho
1312   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1313   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@digamma}{\um@usv@bfdigamma}
1314 }

```

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

```

1315 \cs_new:Npn \um_config_mathbffrak_Latin: {
1316   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bffra
1317 }
1318 \cs_new:Npn \um_config_mathbffrak_latin: {
1319   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bffra
1320   \um_set_mathalphabet_char:Nnn{\mathbffrak}{\um@usv@ith} {"1D58D}
1321 }

```

9.2.5 Bold script or calligraphic: \mathbfbfscr

```

1322 \cs_new:Npn \um_config_mathbfscr_Latin: {
1323   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrL
1324 }
1325 \cs_new:Npn \um_config_mathbfscr_latin: {
1326   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrL
1327   \um_set_mathalphabet_char:Nnn{\mathbfscr}{\um@usv@ith} {"1D4F1}
1328 }

```

9.2.6 Bold sans serif: \mathbfbfsf

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

```

1329 \cs_new:Npn \um_config_mathbfsf_num: {
1330   \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1331 }
1332 \cs_new:Npn \um_config_mathbfsf_Latin: {
1333   \bool_if:NTF \g_um_sfliteral_bool {
1334     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin}{\um@usv@bfsfupLatin}
1335     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itLatin}{\um@usv@bfsfitLatin}
1336   }{
1337     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um_bfsf_Lati
1338   }
1339 }
1340 \cs_new:Npn \um_config_mathbfsf_latin: {
1341   \bool_if:NTF \g_um_sfliteral_bool {
1342     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@uplatin}{\um@usv@bfsfuplatin}
1343     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itlatin}{\um@usv@bfsfitlatin}
1344     \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
1345   }{
1346     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um_bfsf_Lati
1347     \bool_if:NTF \g_um_upsans_bool {
1348       \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D5F5}
1349     }{
1350       \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
1351     }
1352   }
1353 }
1354 \cs_new:Npn \um_config_mathbfsf_Greek: {
1355   \bool_if:NTF \g_um_sfliteral_bool {
1356     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek}{\um@usv@bfsfupGreek}
1357     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itGreek}{\um@usv@bfsfitGreek}
1358     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{{"1D767}
1359     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{{"1D7A1}
1360     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla}{\um@usv@bfsfNabla}
1361     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itNabla}{\um@usv@bfsfitNabla}
1362   }{

```

```

1363 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um_bfsf_Gree
1364 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{\um_bfsfNabla_up
1365 \bool_if:NTF \g_um_upsans_bool {
1366 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{1D767}
1367 }{
1368 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{1D7A1}
1369 }
1370 }
1371 }
1372 \cs_new:Npn \um_config_mathbfsf_greek: {
1373 \bool_if:NTF \g_um_sfliteral_bool {
1374 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek}{\um@usv@bfsfupgreek}
1375 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itgreek}{\um@usv@bfsfitgreek}
1376 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial}{\um@usv@bfsfpartial}
1377 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon}{1D78A}
1378 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta}{1D78B}
1379 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa}{1D78C}
1380 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi}{1D78D}
1381 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho}{1D78E}
1382 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi}{1D78F}
1383 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itpartial}{\um@usv@bfsfitpartial}
1384 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarepsilon}{1D7C4}
1385 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvartheta}{1D7C5}
1386 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarkappa}{1D7C6}
1387 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarphi}{1D7C7}
1388 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarrho}{1D7C8}
1389 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarpi}{1D7C9}
1390 }{
1391 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um_bfsf_gree
1392 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial,\um@usv@itpartial}{\um_bfsfpart
1393 \bool_if:NTF \g_um_upsans_bool {
1394 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7
1395 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1396 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1397 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1398 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{1D78E}
1399 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1400 }{
1401 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7
1402 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{1D7C5}
1403 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{1D7C6}
1404 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{1D7C7}
1405 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{1D7C8}
1406 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{1D7C9}
1407 }
1408 }

```

1409 }

9.2.7 Bold upright sans serif: `\mathbfsfup`

```
1410 \cs_new:Npn \um_config_mathbfsfup_num: {
1411   \um_set_mathalphabet_numbers:Nnn{\mathbfsfup}{\um@usv@num}{\um@usv@bfsfnum}
1412 }
1413 \cs_new:Npn \um_config_mathbfsfup_Latin: {
1414   \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfu
1415 }
1416 \cs_new:Npn \um_config_mathbfsfup_latin: {
1417   \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfu
1418   \um_set_mathalphabet_char:Nnn \mathbfsfup {\um@usv@ith} {"1D5F5}
1419 }
1420 \cs_new:Npn \um_config_mathbfsfup_Greek: {
1421   \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfu
1422   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varTheta,\um@usv@itvarTheta}{1D767}
1423   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@Nabla,\um@usv@itNabla}{1D76F}
1424 }
1425 \cs_new:Npn \um_config_mathbfsfup_greek: {
1426   \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfu
1427   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@partial,\um@usv@itpartial}{1D789}
1428   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D78A}
1429   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1430   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1431   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1432   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varrho,\um@usv@itvarrho}{1D78E}
1433   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1434 }
```

9.2.8 Bold italic sans serif: `\mathbfsfit`

```
1435 \cs_new:Npn \um_config_mathbfsfit_num: {
1436   \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@num}{\um@usv@bfsfnum}
1437 }
1438 \cs_new:Npn \um_config_mathbfsfit_Latin: {
1439   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfi
1440 }
1441 \cs_new:Npn \um_config_mathbfsfit_latin: {
1442   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfi
1443   \um_set_mathalphabet_char:Nnn \mathbfsfit {\um@usv@ith} {"1D65D}
1444 }
1445 \cs_new:Npn \um_config_mathbfsfit_Greek: {
1446   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfi
1447   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{1D7A1}
1448   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNa
1449 }
1450 \cs_new:Npn \um_config_mathbfsfit_greek: {
```

```

1451 \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfi
1452 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsf
1453 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{ "1D7C4
1454 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{ "1D7C5}
1455 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{ "1D7C6}
1456 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{ "1D7C7}
1457 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{ "1D7C8}
1458 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{ "1D7C9}
1459 }

```

9.3 Definitions of the math symbols

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

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

```

1460 \begingroup
1461 \char_make_other:N \^
1462 \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1463 \lowercase{
1464 \scantokens{\global\let#1=^^^#2}
1465 }
1466 }

```

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

```

1467 \gdef\um@scanactivedef"#1\@nil#2{
1468 \lowercase{
1469 \tl_rescan:nn{
1470 \ExplSyntaxOn
1471 \char_make_math_superscript:N\^
1472 }{
1473 \global\def^^^#1{#2}
1474 }
1475 }
1476 }
1477 \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`.


```

1478 \begingroup
1479 \def\UnicodeMathSymbol#1#2#3#4{
1480   \um@scancharlet#2=#1\@nil
1481 }
1482 \char_make_other:N \#
1483 \@input{unicode-math-table.tex}
1484 \endgroup
Fix \backslash:
1485 \group_begin:
1486 \lccode`*=`\\
1487 \char_make_escape:N \|
1488 \char_make_other:N \\
1489 |lowercase{
1490 |group_end:|let|backslash=*}

```

10 Epilogue

Lots of little things to tidy up.

10.0.1 Primes

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

```

U+2032: PRIME (\primesingle): x'
U+2033: DOUBLE PRIME (\primedouble): x''
U+2034: TRIPLE PRIME (\primetriple): x'''
U+2057: QUADRUPLE PRIME (\primequadruple): x''''

```

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

```

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

```

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

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

- Prime encountered; pcount=1.

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

```

1491 \muskip_new:N \g_um_primekern_muskip
1492 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1493 \num_new:N \l_um_primecount_num

1494 \cs_new:Nn \um_nprimes:n {
1495   ^{
1496     \primesingle
1497     \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1498   }
1499 }

1500 \cs_new:Nn \um_nprimes_select:n {
1501   \prg_case_int:nnn {#1}{
1502     {1} { ^{\primesingle} }
1503     {2} {
1504       \um_glyph_if_exist:nTF {"2033} { ^{\primedouble} } {\um_nprimes:n {#1}}
1505     }
1506     {3} {
1507       \um_glyph_if_exist:nTF {"2034} { ^{\primetriples} } {\um_nprimes:n {#1}}
1508     }
1509     {4} {
1510       \um_glyph_if_exist:nTF {"2057} { ^{\primequadruple} } {\um_nprimes:n {#1}}
1511     }
1512   }{
1513     \um_nprimes:n {#1}
1514   }
1515 }

```

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.

```

1516 \cs_new:Nn \um_scanprime: {
1517   \num_zero:N \l_um_primecount_num
1518   \um_scanprime_collect:
1519 }

```

```

1520 \cs_new:Nn \um_scanprime_collect: {
1521   \num_incr:N \l_um_primecount_num
1522   \peek_meaning_remove:NTF ' {
1523     \um_scanprime_collect:
1524   }{
1525     \peek_meaning_remove:NTF \um_scanprime: {
1526       \um_scanprime_collect:
1527     }{
1528       \peek_meaning_remove:NTF ^^^^2032 {
1529         \um_scanprime_collect:
1530       }{
1531         \um_nprimes_select:n {\l_um_primecount_num}
1532       }
1533     }
1534   }
1535 }

1536 \cs_set_eq:NN \prime \um_scanprime:
1537 \group_begin:
1538   \char_make_active:N \
1539   \char_make_active:n {"2032}
1540   \cs_gset_eq:NN ' \um_scanprime:
1541   \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1542 \group_end:

```

10.0.2 Unicode radicals

Undo the damage made to `\sqrt`:

```

1543 \DeclareRobustCommand\sqrt{\ifnextchar[\@sqrt\sqrtsign}

```

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

`#2` : Leading superscript for the sqrt sign

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

```

1544 \def\r@@t#1#2{
1545   \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
1546   \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1547   \raise \dimexpr(
1548     \um@fontdimen@percent{65}{\um@font}\ht\z@-
1549     \um@fontdimen@percent{65}{\um@font}\dp\z@
1550   )\relax
1551   \copy \rootbox
1552   \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1553   \box \z@
1554 }

```

10.0.3 Unicode sub- and super-scripts

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

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

First, the setup of each mathactive char:

```
1555 \prop_new:N \g_um_supers_prop
1556 \prop_new:N \g_um_subs_prop
1557 \cs_generate_variant:Nn \prop_gput:Nnn {Nxn}
1558 \cs_generate_variant:Nn \prop_get:NnN {cxN}
1559 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}
1560
1561 \group_begin:
1562
1563 % Populate a property list with superscript characters; their mean-
1564 % ing as their key,
1565 % for reasons that will become apparent soon, and their replace-
1566 % ment as each key's value.
1567 % Then make the superscript active and bind it to the scanning function.
1568 %
1569 % \cs{scantokens} makes this process much simpler since we can acti-
1570 % vate the char
1571 % and assign its meaning in one step.
1572 \cs_set:Nn \um_setup_active_superscript:nn {
1573   \prop_gput:Nxn \g_um_supers_prop {\meaning #1} {#2}
1574   \char_make_active:n `{#1}
1575   \global\XeTeXmathcodenum`#1 = "1FFFFF \scan_stop:
1576   \scantokens{
1577     \cs_gset:Npn #1 {
1578       \tl_set:Nn \l_um_ss_chain_tl {#2}
1579       \cs_set_eq:NN \um_sub_or_super:n \sp
1580       \tl_set:Nn \l_um_tmpa_tl {supers}
1581       \um_scan_sscript:
1582     }
1583   }
1584 }
1585
1586 \um_setup_active_superscript:nn {^^^2070} {0}
1587 \um_setup_active_superscript:nn {^^^00b9} {1}
1588 \um_setup_active_superscript:nn {^^^00b2} {2}
1589 \um_setup_active_superscript:nn {^^^00b3} {3}
```

```

1587 \um_setup_active_superscript:nn {^^^^2074} {4}
1588 \um_setup_active_superscript:nn {^^^^2075} {5}
1589 \um_setup_active_superscript:nn {^^^^2076} {6}
1590 \um_setup_active_superscript:nn {^^^^2077} {7}
1591 \um_setup_active_superscript:nn {^^^^2078} {8}
1592 \um_setup_active_superscript:nn {^^^^2079} {9}
1593 \um_setup_active_superscript:nn {^^^^207a} {+}
1594 \um_setup_active_superscript:nn {^^^^207b} {-}
1595 \um_setup_active_superscript:nn {^^^^207c} {=}
1596 \um_setup_active_superscript:nn {^^^^207d} {(}
1597 \um_setup_active_superscript:nn {^^^^207e} {)}
1598 \um_setup_active_superscript:nn {^^^^2071} {i}
1599 \um_setup_active_superscript:nn {^^^^207f} {n}
1600
1601 % Ditto above.
1602 \cs_set:Nn \um_setup_active_subscript:nn {
1603   \prop_gput:Nxn \g_um_subs_prop {\meaning #1} {#2}
1604   \char_make_active:n {\`#1}
1605   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1606   \scantokens{
1607     \cs_gset:Npn #1 {
1608       \tl_set:Nn \l_um_ss_chain_tl {#2}
1609       \cs_set_eq:NN \um_sub_or_super:n \sb
1610       \tl_set:Nn \l_um_tmpa_tl {subs}
1611       \um_scan_sscript:
1612     }
1613   }
1614 }
1615
1616 \um_setup_active_subscript:nn {^^^^2080} {0}
1617 \um_setup_active_subscript:nn {^^^^2081} {1}
1618 \um_setup_active_subscript:nn {^^^^2082} {2}
1619 \um_setup_active_subscript:nn {^^^^2083} {3}
1620 \um_setup_active_subscript:nn {^^^^2084} {4}
1621 \um_setup_active_subscript:nn {^^^^2085} {5}
1622 \um_setup_active_subscript:nn {^^^^2086} {6}
1623 \um_setup_active_subscript:nn {^^^^2087} {7}
1624 \um_setup_active_subscript:nn {^^^^2088} {8}
1625 \um_setup_active_subscript:nn {^^^^2089} {9}
1626 \um_setup_active_subscript:nn {^^^^208a} {+}
1627 \um_setup_active_subscript:nn {^^^^208b} {-}
1628 \um_setup_active_subscript:nn {^^^^208c} {=}
1629 \um_setup_active_subscript:nn {^^^^208d} {(}
1630 \um_setup_active_subscript:nn {^^^^208e} {)}
1631 \um_setup_active_subscript:nn {^^^^2090} {a}
1632 \um_setup_active_subscript:nn {^^^^2091} {e}

```

```

1633 \um_setup_active_subscript:nn {^^^1d62} {i}
1634 \um_setup_active_subscript:nn {^^^2092} {o}
1635 \um_setup_active_subscript:nn {^^^1d63} {r}
1636 \um_setup_active_subscript:nn {^^^1d64} {u}
1637 \um_setup_active_subscript:nn {^^^1d65} {v}
1638 \um_setup_active_subscript:nn {^^^2093} {x}
1639 \um_setup_active_subscript:nn {^^^1d66} {\beta}
1640 \um_setup_active_subscript:nn {^^^1d67} {\gamma}
1641 \um_setup_active_subscript:nn {^^^1d68} {\rho}
1642 \um_setup_active_subscript:nn {^^^1d69} {\phi}
1643 \um_setup_active_subscript:nn {^^^1d6a} {\chi}
1644
1645 \group_end:
1646
1647 % The scanning command, evident in its purpose:
1648 \cs_new:Nn \um_scan_sscript: {
1649   \um_scan_sscript:TF {
1650     \um_scan_sscript:
1651   }{
1652     \um_sub_or_super:n {\l_um_ss_chain_tl}
1653   }
1654 }
1655
1656 % The main theme here is stolen from the source to the vari-
1657   ous \cs{peek_} functions.
1658 % Consider this function as simply boilerplate:
1659 \cs_new:Nn \um_scan_sscript:TF {
1660   \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
1661   \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
1662   \tl_set:Nx \l_peek_false_tl { \exp_not:n{\group_align_safe_end: #2}}
1663   \group_align_safe_begin:
1664   \peek_after:NN \um_peek_execute_branches_ss:
1665 }
1666
1667 % We do not skip spaces when scanning ahead, and we explicitly wish to
1668 % bail out on encountering a space or a brace.
1669 \cs_new:Npn \um_peek_execute_branches_ss: {
1670   \bool_if:nTF {
1671     \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
1672     \token_if_eq_catcode_p:NN \l_peek_token \c_group_end_token ||
1673     \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
1674   }
1675   { \l_peek_false_tl }
1676   { \um_peek_execute_branches_ss_aux: }
1677 }

```

```

1678 % This is the actual comparison code.
1679 % Because the peeking has already tokenised the next token,
1680 % it's too late to extract its charcode directly. Instead,
1681 % we look at its meaning, which remains a `character' even
1682 % though it is itself math-active. If the character is ever
1683 % made fully active, this will break our assumptions!
1684 %
1685 % If the char's meaning exists as a property list key, we
1686 % build up a chain of sub-/superscripts and iterate. (If not, exit and
1687 % typeset what we've already collected.)
1688 \cs_new:Nn \um_peek_execute_branches_ss_aux: {
1689   \prop_if_in:cxTF
1690     {g_um_\l_um_tmpa_tl _prop}
1691     {\meaning\l_peek_token}
1692   {
1693     \prop_get:cxN
1694       {g_um_\l_um_tmpa_tl _prop}
1695       {\meaning\l_peek_token}
1696     \l_um_tmpb_tl
1697     \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmpb_tl
1698     \l_peek_true_tl
1699   }
1700   {\l_peek_false_tl}
1701 }

```

10.0.4 Synonyms and all the rest

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

```

1702 \def\operator@font{\um_setup_mathup:}
1703 \def\to{\rightarrow}
1704 \def\vec{\overrightarrow}
1705 \def\le{\leq}
1706 \def\ge{\geq}
1707 \def\neq{\neq}

```

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

```

1708 \@ifpackageloaded{amsmath}{
1709   % define their own colon, perhaps I should just steal it.
1710 }{
1711   \cs_set_protected:Npn \colon {
1712     \bool_if:NTF \g_um_literal_colon_bool { : } { \mathpunct{:} }
1713   }
1714 }

```

`\mathcal`

```
1715 \def\mathcal{\mathscr}
```

`\mathrm`

```
1716 \def\mathrm{\mathup}
```

10.0.5 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:

```
1717 \ifpackageloaded{amsmath}{
1718   \tl_remove_in:Nn \@begindocumenthook {
1719     \mathchardef\std@minus\mathcode`\-\relax
1720     \mathchardef\std@equal\mathcode`\=\relax
1721   }
1722 }{}
```

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

```
1723 \ifpackageloaded{amsopn}{
1724   \cs_set:Npn \newmcodes@ {
1725     \mathcode`\'39
1726     \mathcode`\*42
1727     \mathcode`\."613A%
1728     % \ifnum\mathcode`\-=45 \else
1729     %   \mathchardef\std@minus\mathcode`\-\relax
1730     % \fi
1731     \mathcode`\-45
1732     \mathcode`\ /47
1733     \mathcode`\."603A\relax
1734   }
1735 }{}
```

Octothorpe is an odd one:

```
1736 \AtBeginDocument{
1737   \def\#{\mode_if_math:TF{\mathoctothorpe}{\char`\#}}
1738   \def\widehat{\hat}
1739   \def\widetilde{\tilde}
1740 }
```



```

\digamma I might end up just changing these in the table.
\Digamma 1741 \def\digamma{\updigamma}
          1742 \def\Digamma{\upDigamma}

          Overriding amsmath definitions:
1743 \AtBeginDocument{
1744   \def\@cdots{\mathinner{\cdots}}
1745 }

          Interaction with beamer:
1746 \@ifclassloaded{beamer}{
1747   \ifbeamer@suppressreplacements\else
1748     \PackageWarningNoLine{unicode-math}{
1749       Disabling~ beamer's~ math~ setup.^{}}
1750     Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1751   }
1752   \beamer@suppressreplacementstrue
1753 \fi
1754 }{}

          The end.
1755 \ExplSyntaxOff

```

File II

STIX table data extraction

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

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

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

```

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

```

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

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):

Print the actual entry corresponding to the unicode character:

Now replace the stix class abbreviations with their \TeX macro names.

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

Fixing up a couple of things in the STIX table.

A Documenting maths support in the NFSS

In the following, $\langle NFSS\ decl. \rangle$ stands for something like $\{T1\}\{1mr\}\{m\}\{n\}$.

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

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

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

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

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

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

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

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

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

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

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

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

Delimiters and radicals use wrappers around TeX’s `\delimiter`/`\radical` primitives, which are re-designed in XeTeX. The syntax used in L^AT_EX’s NFSS is therefore not so relevant here.

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

`\DeclareMathDelimiter{<symbol>}{<type>}{<sym.font>}{<slot>}{<sym.font>}{<slot>}`

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

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

Accents are not included yet.

Summary For symbols, something like:

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

For characters, something like:

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

File III

X_YTeX math font dimensions

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

<code>\fontdimen</code>	Dimension name	Description
10	<code>SCRIPTPERCENTSCALEDOWN</code>	Percentage of scaling down for script level 1. Suggested value: 80%.
11	<code>SCRIPTSCRIPTPERCENTSCALEDOWN</code>	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	<code>DELIMITEDSUBFORMULAMINHEIGHT</code>	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height \times 1.5.
13	<code>DISPLAYOPERATORMINHEIGHT</code>	Minimum height of n-ary operators (such as integral and summation) for formulas in display mode.
14	<code>MATHLEADING</code>	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 (<code>os2.sTypoAscender</code> + <code>os2.sTypoLineGap</code> – <code>MathLeading</code>) or with ink going below <code>os2.sTypoDescender</code> 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	FLATTENEDACCENTBASE-HEIGHT	Maximum (ink) height of accent base that does not require flattening the accents. Suggested: cap height of the font (os2.sCapHeight).
18	SUBSCRIPTSHIFTDOWN	The standard shift down applied to subscript elements. Positive for moving in the downward direction. Suggested: os2.ySubscriptYOffset.
19	SUBSCRIPTTOPMAX	Maximum allowed height of the (ink) top of subscripts that does not require moving subscripts further down. Suggested: $/5$ x-height.
20	SUBSCRIPTBASELINEDROPMIN	Minimum allowed drop of the baseline of subscripts relative to the (ink) bottom of the base. Checked for bases that are treated as a box or extended shape. Positive for subscript baseline dropped below the base bottom.
21	SUPERSCRIPSHIFTUP	Standard shift up applied to superscript elements. Suggested: os2.ySuperscriptYOffset.
22	SUPERSCRIPSHIFTUPCRAMPED	Standard shift of superscripts relative to the base, in cramped style.
23	SUPERSCRIPBOTTOMMIN	Minimum allowed height of the (ink) bottom of superscripts that does not require moving subscripts further up. Suggested: $\frac{1}{4}$ x-height.
24	SUPERSCRIPBASELINEDROP-MAX	Maximum allowed drop of the baseline of superscripts relative to the (ink) top of the base. Checked for bases that are treated as a box or extended shape. Positive for superscript baseline below the base top.
25	SUBSUPERSCRIPGAPMIN	Minimum gap between the superscript and subscript ink. Suggested: $4 \times$ default rule thickness.

\fontdimen	Dimension name	Description
26	SUPERSCRIPBTOTTOMMAX- 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	LOWERLIMITBASELINE DROP- 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: $3 \times$ 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: $7 \times$ 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	FRACTIONDENOMDISPLAY- STYLEGAPMIN	Minimum tolerated gap between the (ink) top of the denominator and the ink of the fraction bar in display style. Suggested: 3×default rule thickness.
51	SKEWEDFRACTION- HORIZONTALGAP	Horizontal distance between the top and bottom elements of a skewed fraction.
52	SKEWEDFRACTIONVERTICAL- GAP	Vertical distance between the ink of the top and bottom elements of a skewed fraction.
53	OVERBARVERTICALGAP	Distance between the overbar and the (ink) top of the base. Suggested: 3×default rule thickness.
54	OVERBARRULETHICKNESS	Thickness of overbar. Suggested: default rule thickness.
55	OVERBAREXTRAASCENDER	Extra white space reserved above the overbar. Suggested: default rule thickness.
56	UNDERBARVERTICALGAP	Distance between underbar and (ink) bottom of the base. Suggested: 3×default rule thickness.
57	UNDERBARRULETHICKNESS	Thickness of underbar. Suggested: default rule thickness.
58	UNDERBAREXTRADESCENDER	Extra white space reserved below the underbar. Always positive. Suggested: default rule thickness.
59	RADICALVERTICALGAP	Space between the (ink) top of the expression and the bar over it. Suggested: 1¼ default rule thickness.
60	RADICALDISPLAYSTYLE- VERTICALGAP	Space between the (ink) top of the expression and the bar over it. Suggested: default rule thickness + ¼ x-height.
61	RADICALRULETHICKNESS	Thickness of the radical rule. This is the thickness of the rule in designed or constructed radical signs. Suggested: default rule thickness.
62	RADICALEXTRAASCENDER	Extra white space reserved above the radical. Suggested: RadicalRuleThickness.
63	RADICALKERNBEFOREDEGREE	Extra horizontal kern before the degree of a radical, if such is present. Suggested: 5/18 of em.

\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%.

Index

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

Symbols	
\"	17
\#	1482, 1737
\'	1538, 1725
*	502, 1486, 1726
\-	501, 1719, 1728, 1729, 1731
\.	1727
\/	551, 1732
\:	504, 1733
\::	788
\::f	788
\::n	788
\<	555
\=	1720
\>	556
\@DeclareMathDelimiter	343
\@DeclareMathSizes	308
\@backslashchar	980
\@beginofdocumenthook	1718
\@cclvi	368
\@cdots	1744
\@elt	659–663, 666, 670, 672
\@empty	419, 420, 461, 588, 704, 711, 726, 747, 752
\@ifclassloaded	1746
\@ifnextchar	1543
\@ifpackageloaded	1708, 1717, 1723
\@ii	710, 711, 713, 715, 718, 723
\@input	478, 1483
\@marker	723, 742
\@nil	377, 532, 723, 736–739, 1462, 1467, 1480
\@preamblecmds	352
\@sqrt	1543
\@tempa	136, 183, 215, 241, 259, 277, 285, 292, 434, 450, 702, 714, 738, 740, 752
\@tempb	136, 137, 183, 184, 215, 216, 241, 242, 259, 260, 277, 278, 285, 286, 292, 293, 702, 703, 741, 742, 747
\@tempswafalse	712
\@tempswatru	716, 719, 744, 749, 754, 759
\@um@bfliteraltrue	178, 203
\@um@bfupGreekfalse	142, 185
\@um@bfupGreektrue	155, 168, 191, 197
\@um@bfupLatinfalse	145, 187
\@um@bfupLatintrue	158, 171, 193, 199
\@um@bfupgreekfalse	143, 156, 186, 192
\@um@bfupgreektrue	169, 198
\@um@bfuplatinfalse	146, 188
\@um@bfuplatintrue	159, 172, 194, 200
\@um@fontspec@featuretrue	421
\@um@literaltrue	177
\@um@ot@math@true	453
\@um@upGreekfalse	138
\@um@upGreektrue	151, 164
\@um@upLatinfalse	140, 153
\@um@upLatintrue	166
\@um@upNablafalse	147, 245
\@um@upNablatrue	160, 173, 243
\@um@upgreekfalse	139, 152
\@um@upgreektrue	165
\@um@uplatinfalse	141, 154, 167
\@um@uppartialfalse	144, 157, 189, 195, 263
\@um@uppartialtrue	170, 201, 261
\@xDeclareMathDelimiter	344
\@xxDeclareMathDelimiter	342
\\	10–13, 17, 23–33, 1486, 1488
\^	33, 1461, 1471
\	1487
Numbers	
\0	23
\sqcup	17–20, 23–32
A	
\A	24
\a	25
\addnolimits	<u>665</u>

\addtoversion	315	\clist_map_break:	971
\alloc@	368	\clist_map_inline:Nn	645
\Alpha	815	\clist_map_inline:nn	
\alpha	840	537, 640, 647, 968, 974
\alpha@elt	319	\clist_map_variable:NNn	710
\alpha@list	318	\clist_map_variable:nNn ..	767, 790, 796
\AtBeginDocument	813, 1736, 1743	\colon	1711
\awint	661	\copy	1551
B		\cs	1567, 1656
\B	1096	\cs_generate_variant:Nn ...	1557–1559
\beamer@suppressreplacementstrue	1752	\cs_gset:cpn	382, 392
\begingroup	374, 669, 1460, 1478	\cs_gset:Npn ..	395, 400, 1462, 1574, 1607
\Beta	816	\cs_gset:Npx	405
\beta	841, 1639	\cs_gset_eq:NN	1540, 1541
\bgroup	1010	\cs_if_exist:cF	1005
\bool_if:NF	503	\cs_if_exist:cT	993
\bool_if:NTF	225, 845, 864, 870, 875, 1125, 1133, 1139, 1333, 1341, 1347, 1355, 1365, 1373, 1393, 1712	\cs_new:Nn 486, 500, 531, 536, 541, 544, 550, 581, 584, 789, 795, 804, 807, 810, 945, 967, 992, 1004, 1494, 1500, 1516, 1520, 1648, 1658, 1688	
\bool_if:nTF	1669	\cs_new:Npn	1019, 1022, 1026, 1031, 1041, 1044, 1048, 1053, 1063, 1067, 1077, 1080, 1087, 1094, 1105, 1112, 1120, 1124, 1132, 1146, 1149, 1152, 1156, 1159, 1162, 1165, 1169, 1172, 1175, 1179, 1182, 1190, 1204, 1223, 1262, 1265, 1268, 1272, 1277, 1288, 1291, 1294, 1298, 1304, 1315, 1318, 1322, 1325, 1329, 1332, 1340, 1354, 1372, 1410, 1413, 1416, 1420, 1425, 1435, 1438, 1441, 1445, 1450, 1668
\bool_new:N	22, 212, 213, 284	\cs_set:cpn	1006
\bool_set_false:N		\cs_set:Nn	206, 224, 248, 266, 414, 492, 495, 587, 639, 644, 766, 776, 779, 782, 785, 880, 891, 901, 908, 917, 926, 997, 1569, 1602
... 148, 149, 175, 180, 217, 279, 289		\cs_set:Npn	788, 1724
\bool_set_true:N		\cs_set:Npx	430
161, 162, 174, 179, 219, 221, 281, 287		\cs_set_eq:cN	999
\box	1553	\cs_set_eq:NN	
C		464–467, 471–474, 1536, 1576, 1609	
\C	1069, 1114	\cs_set_protected:cpn	1009
\c_group_begin_token	1670	\cs_set_protected:Npn	1711
\c_group_end_token	1671	\cs_to_str:N	377, 382, 392, 654
\c_peek_true_remove_next_tl ...	1660		
\c_space_token	1672		
\cdots	1744		
\cdp@elt	306		
\cdp@list	305		
\char	1737		
\char_make_active:N	1538		
\char_make_active:n 375, 1539, 1571, 1604			
\char_make_escape:N	1487		
\char_make_math_superscript:N ..	1471		
\char_make_other:N ..	1461, 1482, 1488		
\chardef	368		
\Chi	837		
\chi	866, 1643		
\cirfnint	661		

<code>\csname</code>	371, 377, 380, 383, 415, 422, 546, 582, 585, 969, 975, 1013		
D			
<code>\D</code>	1088	<code>\else:</code>	1002
<code>\d</code>	1089	<code>\encodingdefault</code>	477
<code>\DeclareDocumentCommand</code>	132, 417	<code>\endcsname</code>	371, 377, 380, 383, 415, 422, 546, 582, 585, 969, 975, 1013
<code>\DeclareMathAccent</code>	336	<code>\endgroup</code>	378, 675, 1477, 1484
<code>\DeclareMathAlphabet</code>	332	<code>\Epsilon</code>	819
<code>\DeclareMathDelimiter</code>	341	<code>\epsilon</code>	844
<code>\DeclareMathRadical</code>	347	<code>\Eta</code>	821
<code>\DeclareMathSizes</code>	307	<code>\eta</code>	848
<code>\DeclareMathSymbol</code>	338	<code>\etex_iffontchar:D</code>	1002
<code>\DeclareMathVersion</code>	312, 424	<code>\ExecuteOptionsX</code>	301
<code>\DeclareRobustCommand</code>	1543	<code>\exp_args:Nnff</code>	788, 791, 798
<code>\DeclareSymbolFont</code>	325, 476	<code>\exp_args:No</code>	641, 648
<code>\DeclareSymbolFontAlphabet</code>	349	<code>\exp_not:N</code>	431
<code>\DeclareSymbolFontAlphabet@</code>	350	<code>\exp_not:n</code>	1659, 1661
<code>\def</code>	23–131, 354, 357, 368, 370, 372, 423, 658, 666, 668, 670, 677, 679, 736, 738–741, 815–844, 847–863, 866–869, 872–874, 877, 878, 1473, 1479, 1544, 1702–1707, 1715, 1716, 1737–1739, 1741, 1742, 1744	<code>\exp_not:V</code>	431
<code>\define@choicekey</code>	135, 183, 214, 241, 259, 277, 285, 292, 702	<code>\expandafter</code>	371, 379, 384, 386, 390, 666, 713, 715, 718, 723, 737, 738, 742, 1012, 1013
<code>\define@cmdkey</code>	698–701	<code>\ExplSyntaxOff</code>	1755
<code>\define@key</code>	681	<code>\ExplSyntaxOn</code>	6, 1470
<code>\define@mathalphabet</code>	313	F	
<code>\define@mathgroup</code>	314	<code>\F</code>	1098
<code>\Delta</code>	818	<code>\f@size</code>	422
<code>\delta</code>	843	<code>\fi</code>	181, 204, 207–210, 222, 246, 257, 264, 275, 282, 290, 299, 365, 366, 387, 408–412, 460, 475, 514, 529, 597, 605, 610, 617, 634, 635, 637, 650, 673, 689, 705, 720, 721, 724, 730, 732, 733, 745, 750, 755, 760–764, 906, 915, 924, 943, 1188, 1201, 1202, 1220, 1221, 1258, 1260, 1730, 1753
<code>\Digamma</code>	<u>1741</u>	<code>\fi:</code>	1002
<code>\digamma</code>	<u>1741</u>	<code>\fint</code>	661
<code>\dimexpr</code>	355, 452, 1547	<code>\font</code>	451
<code>\do</code>	352	<code>\fontdimen</code>	355, 452, 1546, 1552
<code>\dorestore@version</code>	322	<code>\fontname</code>	983
<code>\dp</code>	1549	G	
E		<code>\g</code>	1108
<code>\E</code>	1097	<code>\g@addto@macro</code>	727, 729
<code>\e</code>	1090, 1107		
<code>\edef</code>	434, 737, 738		
<code>\egroup</code>	1007, 1012		
<code>\else</code>	207–210, 253, 271, 360, 363, 389, 393, 398, 403, 406, 454, 468, 511, 524, 592, 600,		

<code>\kappa</code>	851	<code>\mathbfit</code>	1263, 1266, 1269, 1270, 1273–1275, 1278–1286
<code>\kern</code>	1546, 1552	<code>\mathbfscr</code>	1323, 1326, 1327
L			
<code>\L</code>	1101	<code>\mathbfsf</code>	1330, 1334, 1335, 1337, 1342–1344, 1346, 1348, 1350, 1356–1361, 1363, 1364, 1366, 1368, 1374–1389, 1391, 1392, 1394–1399, 1401–1406
<code>\l_peek_false_tl</code>	1661, 1674, 1700	<code>\mathbfsfit</code>	1436, 1439, 1442, 1443, 1446–1448, 1451–1458
<code>\l_peek_token</code> ...	1670–1672, 1691, 1695	<code>\mathbfsfup</code>	1411, 1414, 1417, 1418, 1421–1423, 1426–1433
<code>\l_peek_true_aux_tl</code>	1659	<code>\mathbfup</code>	1289, 1292, 1295, 1296, 1299–1302, 1305–1313
<code>\l_peek_true_tl</code>	1660, 1698	<code>\mathbin</code>	501, 502
<code>\l_um_inc_num</code>	797, 799, 800	<code>\mathcal</code>	<u>1715</u>
<code>\l_um_incr_num</code>	768, 770, 772	<code>\mathchar@type</code>	348, 381, 395, 397, 400, 402, 405, 407, 415, 545
<code>\l_um_input_num</code>	767, 770, 790, 792, 796, 799	<code>\mathchardef</code>	1719, 1720, 1729
<code>\l_um_primecount_num</code>	1493, 1517, 1521, 1531	<code>\mathclose</code>	399
<code>\l_um_script_features_tl</code> ...	425, 440	<code>\mathcode</code>	376, 1719, 1720, 1725–1729, 1731–1733
<code>\l_um_script_font_tl</code>	427, 439	<code>\mathfrak</code>	1113–1118, 1121, 1122
<code>\l_um_ss_chain_tl</code> 1575, 1608, 1652, 1697		<code>\mathgroup</code>	368
<code>\l_um_sscript_features_tl</code> ..	426, 444	<code>\mathinner</code>	1744
<code>\l_um_sscript_font_tl</code>	428, 443	<code>\mathit</code>	1042, 1045, 1046, 1049–1051, 1054–1061
<code>\l_um_tmpa_tl</code> ...	1577, 1610, 1690, 1694	<code>\mathoctothorpe</code>	1737
<code>\l_um_tmpb_tl</code>	1696, 1697	<code>\mathop</code>	373
<code>\Lambda</code>	825	<code>\mathopen</code>	394, 679
<code>\lambda</code>	852	<code>\mathord</code>	507–510, 512, 513, 516–523, 525–528, 542
<code>\lccode</code>	1486	<code>\mathpunct</code>	1712
<code>\le</code>	1705	<code>\mathrel</code>	504
<code>\left</code>	<u>678</u>	<code>\mathrm</code>	<u>1716</u>
<code>\left@primitive</code>	678, 679	<code>\mathscr</code> ...	1095–1103, 1106–1110, 1715
<code>\leq</code>	1705	<code>\mathsf</code>	1126, 1127, 1129, 1134–1136, 1138, 1140, 1142, 1147
<code>\let</code>	369, 418–420, 678, 704, 1464	<code>\mathsfbit</code>	1160, 1163, 1166, 1167
<code>\lowercase</code>	1463, 1468	<code>\mathsfup</code>	1150, 1153, 1154, 1157
<code>\lowint</code>	663	<code>\mathhtt</code>	1170, 1173, 1176, 1177
M			
<code>\M</code>	1102	<code>\mathup</code>	1020, 1023, 1024, 1027–1029, 1032–1039, 1716
<code>\m@th</code>	1545	<code>\mddefault</code>	477
<code>\mathaccent</code>	404	<code>\meaning</code>	1570, 1603, 1691, 1695
<code>\mathalpha</code>	655, 771	<code>\mitAlpha</code>	815
<code>\mathbb</code>	1064, 1065, 1068–1075, 1078, 1081–1085		
<code>\mathbbit</code>	1088–1092		
<code>\mathbf</code>	1180, 1184, 1185, 1187, 1192–1194, 1196, 1198, 1200, 1205, 1207–1212, 1214, 1215, 1217, 1219, 1224, 1226–1241, 1243, 1245–1250, 1252–1257, 1259		
<code>\mathbffrak</code>	1316, 1319, 1320		

O	
\o	1109
\oiint	659
\oint	659
\oint	659
\ointctrclockwise	660
\Omega	839
\omega	868
\Omicron	829
\omicron	856
\operator@font	1702
\or	150, 163, 176, 190, 196, 202, 218, 220, 244, 262, 280, 288, 295, 297
\overrightarrow	1704
P	
\P	1072
\PackageError	685
\PackageInfo	463
\PackageWarningNoLine ..	455, 978, 1748
\peek_after:NN	1663
\peek_meaning_remove:NTF	1522, 1525, 1528
\Phi	836
\phi	863, 1642
\Pi	830
\pi	857
\pointint	662
\prg_case_int:nnn	1501
\prg_do_nothing:	999
\prg_new_conditional:Nnn	1001
\prg_replicate:nn	1497
\prg_return_false:	1002
\prg_return_true:	1002
\prg_stepwise_variable:nnnNn	768, 797
\prime	1536
\primedouble	1504
\primequadruple	1510
\primesingle	542, 1496, 1497, 1502
\primetripel	1507
\process@table	323
\ProcessOptionsX	302
\prop_get:cxN	1693
\prop_get:NnN	1558
\prop_gput:Nnn	1557
\prop_gput:Nxn	1570, 1603
\prop_if_in:cxTF	1689
\prop_if_in:NnTF	1559
\prop_new:N	1555, 1556
\protect	688
\ProvidesPackage	1
\Psi	838
\psi	867
Q	
\Q	1073
R	
\R	1074, 1103, 1117
\r@@t	<u>1544</u>
\raise	1547
\relax	137, 184, 216, 242, 260, 278, 286, 293, 355, 373, 376, 381, 392, 394–397, 399–402, 404, 405, 407, 415, 418, 451, 452, 646, 703, 715, 718, 742, 743, 748, 753, 757, 758, 770, 772, 792, 799, 800, 1550, 1719, 1720, 1729, 1733
\removenolimits	<u>668</u>
\RequirePackage	3–5
\restore@mathversion	320
\Rho	831
\rho	858, 1641
\rightarrow	1703
\rootbox	1551
\rppolint	661
S	
\sb	1609
\scan_stop: ..	547, 548, 1002, 1572, 1605
\scantokens	1464, 1573, 1606
\scpolint	662
\scriptscriptstyle	361
\scriptstyle	358
\set@@mathdelimiter	346
\set@mathaccent	337
\set@mathchar	339
\set@mathdelimiter	345
\set@mathsymbol	340
\setbox	1545
\setkeys	133, 429, 431
\SetMathAlphabet	334
\SetMathAlphabet@	335
\setmathfont	<u>417</u> , 688

\SetSymbolFont	329	\um@firstchar	713, 738
\SetSymbolFont@	330	\um@firstof	736–738
\sf@size	438, 442	\um@font	359, 362, 451, 452, 983, 1002, 1546, 1548, 1549, 1552
\Sigma	833	\um@fontdimen@percent	354, 359, 362, 1548, 1549
\sigma	860	\um@mathsymbol	370, 493
\sp	1576	\um@mversion	423, 424
\space	978, 1013	\um@nolimits	384, 658, 666, 674
\sqint	662	\um@parse@range	723, 739
\sqrt	677, 1543	\um@parse@term	496, 532, 709
\sqrtsign	1543, 1545	\um@radicals	390, 677
\std@equal	1720	\um@resolve@greek	813
\std@minus	1719, 1729	\um@scaled@apply	357, 1546, 1552
\stepcounter	469	\um@scanactivedef	377, 1460
\string	380, 383, 737, 738	\um@scancharlet	1460, 1480
\strip@pt	355	\um@set@mathsymbol	371, 372
\sumint	660	\um@usv@bbLatin	33, 1068
T		\um@usv@bblatin	34, 1064
\Tau	834	\um@usv@bbnum	32, 1078
\tau	861	\um@usv@bfDigamma	95, 1205, 1302
\tf@size	437, 438	\um@usv@bfdigamma	102, 1224, 1313
\Theta	822	\um@usv@bffrakLatin	66, 1316
\theta	849	\um@usv@bffraklatin	67, 1319
\theum@fam	470	\um@usv@bfGreek	60
\thinmuskip	1492	\um@usv@bfgreek	61
\tilde	1739	\um@usv@bfitGreek	64, 207, 612, 615, 898, 1208, 1273
\tl_map_inline:nn	303	\um@usv@bfitgreek	65, 208, 619, 627, 899, 1227, 1279
\tl_put_right:cn	654	\um@usv@bfith	112, 1194, 1200
\tl_put_right:Nv	1697	\um@usv@bfitLatin	62, 209, 602, 604, 894, 1185, 1266, 1278
\tl_remove_in:Nn	352, 1718	\um@usv@bfitlatin	63, 210, 607, 609, 895, 1193, 1269
\tl_rescan:nn	1469	\um@usv@bfitNabla	123, 255, 517, 525, 1212, 1275
\tl_set:Nn	226–231, 233–238, 250–252, 254–256, 268–270, 272–274, 294, 296, 298, 425–428, 462, 987–991, 1575, 1577, 1608, 1610	\um@usv@bfitnum	53
\tl_set:Nx	207–210, 470, 1659, 1661	\um@usv@bfitpartial	129, 273, 521, 527, 1235, 1280
\tl_set_eq:NN	1660	\um@usv@bfitvarepsilon	114, 620, 628, 1236, 1252, 1281
\tl_use:c	981	\um@usv@bfitvarkappa	116, 622, 630, 1238, 1254, 1283
\to	1703	\um@usv@bfitvarphi	117, 623, 631, 1239, 1255, 1284
\token_if_eq_catcode_p:NN	1670, 1671		
\token_if_eq_meaning_p:NN	1672		
U			
\um@addto@mathmap	641, 648, 653		
\um@backslash	714, 737		
\um@char@num@range	420, 645, 726, 727, 729		
\um@char@range	419, 461, 588, 704, 707, 710		

$\backslash\mathrm{um@usv@bfitvarpi}$	$\backslash\mathrm{um@usv@bfuplatin}$
.... 119, 625, 633, 1241, 1257, 1286	.. 56, 210, 607, 609, 893, 1192, 1295
$\backslash\mathrm{um@usv@bfitvarrho}$	$\backslash\mathrm{um@usv@bfupnum}$
.... 118, 624, 632, 1240, 1256, 1285	52
$\backslash\mathrm{um@usv@bfitvartheta}$	$\backslash\mathrm{um@usv@bfvarepsilon}$
.... 113, 613, 616, 1210, 1219, 1274 96, 620, 628, 1229, 1245, 1307
$\backslash\mathrm{um@usv@bfitvartheta}$	$\backslash\mathrm{um@usv@bfvarkappa}$
.... 115, 621, 629, 1237, 1253, 1282 98, 622, 630, 1231, 1247, 1309
$\backslash\mathrm{um@usv@bflatin}$	$\backslash\mathrm{um@usv@bfvarphi}$
55 99, 623, 631, 1232, 1248, 1310
$\backslash\mathrm{um@usv@bflatin}$	$\backslash\mathrm{um@usv@bfvarpi}$
57 101, 625, 633, 1234, 1250, 1312
$\backslash\mathrm{um@usv@bfNabla}$	$\backslash\mathrm{um@usv@bfvarrho}$
.... 122, 251, 516, 525, 1211, 1301 100, 624, 632, 1233, 1249, 1311
$\backslash\mathrm{um@usv@bfnum}$	$\backslash\mathrm{um@usv@bfvartheta}$
.... 128, 269, 520, 527, 1228, 1306 94, 613, 616, 1209, 1217, 1300
$\backslash\mathrm{um@usv@bfscrLatin}$	$\backslash\mathrm{um@usv@bfvartheta}$
68, 1323 97, 621, 629, 1230, 1246, 1308
$\backslash\mathrm{um@usv@bfscrLatin}$	$\backslash\mathrm{um@usv@Digamma}$
69, 1326	86, 1205, 1302
$\backslash\mathrm{um@usv@bfsfGreek}$	$\backslash\mathrm{um@usv@digamma}$
79	93, 1224, 1313
$\backslash\mathrm{um@usv@bfsfgreek}$	$\backslash\mathrm{um@usv@frakLatin}$
80	37, 1113
$\backslash\mathrm{um@usv@bfsfitGreek}$ 83, 237, 1357, 1446	$\backslash\mathrm{um@usv@fraklatin}$
$\backslash\mathrm{um@usv@bfsfitgreek}$ 84, 238, 1375, 1451	38, 1121
$\backslash\mathrm{um@usv@bfsfitLatin}$ 81, 235, 1335, 1439	$\backslash\mathrm{um@usv@itGreek}$
$\backslash\mathrm{um@usv@bfsfitlatin}$ 82, 236, 1343, 1442	30, 888,
$\backslash\mathrm{um@usv@bfsfitNabla}$	919, 922, 1027, 1049, 1208, 1214,
.... 125, 256, 519, 526, 1361, 1448	1273, 1299, 1357, 1363, 1421, 1446
$\backslash\mathrm{um@usv@bfsfitnum}$	$\backslash\mathrm{um@usv@itgreek}$
72	31,
$\backslash\mathrm{um@usv@bfsfitpartial}$	928, 936, 1032, 1054, 1227, 1243,
.... 131, 274, 523, 528, 1383, 1452	1279, 1305, 1375, 1391, 1426, 1451
$\backslash\mathrm{um@usv@bfsfLatin}$	$\backslash\mathrm{um@usv@ith}$
74	.. 103, 884, 911, 914, 1024, 1046,
$\backslash\mathrm{um@usv@bfsflatin}$	1065, 1110, 1122, 1136, 1140,
76	1142, 1154, 1167, 1177, 1194,
$\backslash\mathrm{um@usv@bfsfNabla}$ 124, 252, 518, 526, 1360	1198, 1200, 1270, 1296, 1320,
$\backslash\mathrm{um@usv@bfsfnum}$... 70, 1330, 1411, 1436	1327, 1344, 1348, 1350, 1418, 1443
$\backslash\mathrm{um@usv@bfsfpartial}$	$\backslash\mathrm{um@usv@itLatin}$
.... 130, 270, 522, 528, 1376 28, 882, 903, 905, 1020,
$\backslash\mathrm{um@usv@bfsfupGreek}$ 77, 230, 1356, 1421	1042, 1068, 1095, 1113, 1127,
$\backslash\mathrm{um@usv@bfsfupgreek}$ 78, 231, 1374, 1426	1129, 1157, 1163, 1173, 1185,
$\backslash\mathrm{um@usv@bfsfupLatin}$ 73, 228, 1334, 1414	1187, 1266, 1278, 1292, 1316,
$\backslash\mathrm{um@usv@bfsfuplatin}$ 75, 229, 1342, 1417	1323, 1335, 1337, 1346, 1414, 1439
$\backslash\mathrm{um@usv@bfsfupnum}$	$\backslash\mathrm{um@usv@itlatin}$
71	29,
$\backslash\mathrm{um@usv@bfupGreek}$	883, 910, 913, 1023, 1045, 1064,
.. 58, 207, 612, 615, 896, 1207, 1299	1106, 1121, 1135, 1138, 1153,
$\backslash\mathrm{um@usv@bfupgreek}$	1166, 1176, 1193, 1196, 1269,
.. 59, 208, 619, 627, 897, 1226, 1305	1295, 1319, 1326, 1343, 1417, 1442
$\backslash\mathrm{um@usv@bfuph}$	$\backslash\mathrm{um@usv@itNabla}$
111, 1198	121, 254,
$\backslash\mathrm{um@usv@bfupLatin}$	508, 512, 1028, 1051, 1212, 1215,
.. 54, 209, 602, 604, 892, 1184, 1292	

1275, 1301, 1361, 1364, 1423, 1448	\um@usv@sfuplatin 44, 1134, 1153
\um@usv@itpartial 127, 272,	\um@usv@sfupnum 40
510, 513, 1033, 1055, 1235, 1259,	\um@usv@ttLatin 49, 1173
1280, 1306, 1383, 1392, 1427, 1452	\um@usv@ttlLatin 50, 1176
\um@usv@itvarepsilon	\um@usv@ttnum 48, 1170
. 105, 929, 937, 1034,	\um@usv@upGreek 26, 886,
1056, 1236, 1245, 1252, 1281,	919, 922, 1027, 1049, 1207, 1214,
1307, 1384, 1394, 1401, 1428, 1453	1273, 1299, 1356, 1363, 1421, 1446
\um@usv@itvarkappa	\um@usv@upgreek 27, 889,
. 107, 931, 939, 1036,	928, 936, 1032, 1054, 1226, 1243,
1058, 1238, 1247, 1254, 1283,	1279, 1305, 1374, 1391, 1426, 1451
1309, 1386, 1396, 1403, 1430, 1455	\um@usv@upLatin
\um@usv@itvarphi 108, 932, 940, 1037, 24, 881, 903, 905, 1020,
1059, 1239, 1248, 1255, 1284,	1042, 1068, 1095, 1113, 1126,
1310, 1387, 1397, 1404, 1431, 1456	1129, 1157, 1163, 1173, 1184,
\um@usv@itvarpi 110, 934, 942, 1039,	1187, 1266, 1278, 1292, 1316,
1061, 1241, 1250, 1257, 1286,	1323, 1334, 1337, 1346, 1414, 1439
1312, 1389, 1399, 1406, 1433, 1458	\um@usv@uplatin 25,
\um@usv@itvarrho 109, 933, 941, 1038,	885, 910, 913, 1023, 1045, 1064,
1060, 1240, 1249, 1256, 1285,	1106, 1121, 1134, 1138, 1153,
1311, 1388, 1398, 1405, 1432, 1457	1166, 1176, 1192, 1196, 1269,
\um@usv@itvarTheta	1295, 1319, 1326, 1342, 1417, 1442
104, 923, 1029, 1050, 1210, 1217,	\um@usv@varepsilon 87, 929, 937, 1034,
1219, 1274, 1300, 1366, 1368, 1422	1056, 1229, 1245, 1252, 1281,
\um@usv@itvartheta	1307, 1377, 1394, 1401, 1428, 1453
. 106, 930, 938, 1035,	\um@usv@varkappa 89, 931, 939, 1036,
1057, 1237, 1246, 1253, 1282,	1058, 1231, 1247, 1254, 1283,
1308, 1385, 1395, 1402, 1429, 1454	1309, 1379, 1396, 1403, 1430, 1455
\um@usv@Nabla 120, 250,	\um@usv@varphi . . 90, 932, 940, 1037,
507, 512, 1028, 1051, 1211, 1215,	1059, 1232, 1248, 1255, 1284,
1275, 1301, 1360, 1364, 1423, 1448	1310, 1380, 1397, 1404, 1431, 1456
\um@usv@num 23,	\um@usv@varpi . . . 92, 934, 942, 1039,
589, 1078, 1147, 1150, 1160, 1170,	1061, 1234, 1250, 1257, 1286,
1180, 1263, 1289, 1330, 1411, 1436	1312, 1382, 1399, 1406, 1433, 1458
\um@usv@partial 126, 268,	\um@usv@varrho . . 91, 933, 941, 1038,
509, 513, 1033, 1055, 1228, 1259,	1060, 1233, 1249, 1256, 1285,
1280, 1306, 1376, 1392, 1427, 1452	1311, 1381, 1398, 1405, 1432, 1457
\um@usv@scrLatin 35, 1095	\um@usv@varTheta
\um@usv@scrLatin 36, 1106	. . 85, 887, 920, 923, 1029, 1050,
\um@usv@sfitLatin . . 46, 233, 1127, 1163	1209, 1217, 1219, 1274, 1300,
\um@usv@sfitLatin . . 47, 234, 1135, 1166	1358, 1359, 1366, 1368, 1422, 1447
\um@usv@sfitnum 41	\um@usv@vartheta 88, 930, 938, 1035,
\um@usv@sflatin 43, 226	1057, 1230, 1246, 1253, 1282,
\um@usv@sflatin 45, 227	1308, 1378, 1395, 1402, 1429, 1454
\um@usv@sfnun 39, 1147, 1150, 1160	\um@zf@feature 680, 692, 695
\um@usv@sfupLatin 42, 1126, 1157	\um_bf_Greek_up_or_it_usv . 207, 1214

<code>\um_bf_greek_up_or_it_usv</code>	208, 1243	<code>\um_config_mathbfsfit_latin:</code>	... 1441
<code>\um_bf_Latin_up_or_it_usv</code>	209, 1187	<code>\um_config_mathbfsfit_num:</code> 1435
<code>\um_bf_latin_up_or_it_usv</code>	210, 1196	<code>\um_config_mathbfsfup_Greek:</code>	... 1420
<code>\um_bfNabla_up_or_it_usv</code>	<code>\um_config_mathbfsfup_greek:</code>	... 1425
	251, 255, 525, 1215	<code>\um_config_mathbfsfup_Latin:</code>	... 1413
<code>\um_bfpartial_up_or_it_usv</code>	<code>\um_config_mathbfsfup_latin:</code>	... 1416
	269, 273, 527, 1259	<code>\um_config_mathbfsfup_num:</code> 1410
<code>\um_bfsf_Greek_up_or_it_usv</code>	<code>\um_config_mathbfup_Greek:</code> 1298
	230, 237, 1363	<code>\um_config_mathbfup_greek:</code> 1304
<code>\um_bfsf_greek_up_or_it_usv</code>	<code>\um_config_mathbfup_Latin:</code> 1291
	231, 238, 1391	<code>\um_config_mathbfup_latin:</code> 1294
<code>\um_bfsf_Latin_up_or_it_usv</code>	<code>\um_config_mathbfup_num:</code> 1288
	228, 235, 1337	<code>\um_config_mathfrak_Latin:</code> 1112
<code>\um_bfsf_latin_up_or_it_usv</code>	<code>\um_config_mathfrak_latin:</code> 1120
	229, 236, 1346	<code>\um_config_mathit_Greek:</code> 1048
<code>\um_bfsfNabla_up_or_it_usv</code>	<code>\um_config_mathit_greek:</code> 1053
	252, 256, 526, 1364	<code>\um_config_mathit_Latin:</code> 1041
<code>\um_bfsfpartial_up_or_it_usv</code>	<code>\um_config_mathit_latin:</code> 1044
	270, 274, 528, 1392	<code>\um_config_mathscr_Latin:</code> 1094
<code>\um_config_mathbb_Latin:</code> 1067	<code>\um_config_mathscr_latin:</code> 1105
<code>\um_config_mathbb_latin:</code> 1063	<code>\um_config_mathsf_Latin:</code> 1124
<code>\um_config_mathbb_misc:</code> 1080	<code>\um_config_mathsf_latin:</code> 1132
<code>\um_config_mathbb_num:</code> 1077	<code>\um_config_mathsf_num:</code> 1146
<code>\um_config_mathbbit_misc:</code> 1087	<code>\um_config_mathsf_Latin:</code> 1162
<code>\um_config_mathbf_Greek:</code> 1204	<code>\um_config_mathsf_Latin:</code> 1165
<code>\um_config_mathbf_greek:</code> 1223	<code>\um_config_mathsfnum:</code> 1159
<code>\um_config_mathbf_Latin:</code> 1182	<code>\um_config_mathsfup_Latin:</code> 1156
<code>\um_config_mathbf_latin:</code> 1190	<code>\um_config_mathsfup_latin:</code> 1152
<code>\um_config_mathbf_num:</code> 1179	<code>\um_config_mathsfup_num:</code> 1149
<code>\um_config_mathbffrak_Latin:</code>	... 1315	<code>\um_config_mathhtt_Latin:</code> 1172
<code>\um_config_mathbffrak_latin:</code>	... 1318	<code>\um_config_mathhtt_latin:</code> 1175
<code>\um_config_mathbfit_Greek:</code> 1272	<code>\um_config_mathhtt_num:</code> 1169
<code>\um_config_mathbfit_greek:</code> 1277	<code>\um_config_mathup_Greek:</code> 1026
<code>\um_config_mathbfit_Latin:</code> 1265	<code>\um_config_mathup_greek:</code> 1031
<code>\um_config_mathbfit_latin:</code> 1268	<code>\um_config_mathup_Latin:</code> 1019
<code>\um_config_mathbfit_num:</code> 1262	<code>\um_config_mathup_latin:</code> 1022
<code>\um_config_mathbfscr_Latin:</code>	... 1322	<code>\um_glyph_if_exist:n</code> 1001
<code>\um_config_mathbfscr_latin:</code>	... 1325	<code>\um_glyph_if_exist:nT</code> 969
<code>\um_config_mathbfsf_Greek:</code> 1354	<code>\um_glyph_if_exist:nTF</code>
<code>\um_config_mathbfsf_greek:</code> 1372		975, 1001, 1504, 1507, 1510
<code>\um_config_mathbfsf_Latin:</code> 1332	<code>\um_init_alphabet:n</code> 467, 997
<code>\um_config_mathbfsf_latin:</code> 1340	<code>\um_make_mathactive:nNN</code> 542, <u>544</u>
<code>\um_config_mathbfsf_num:</code> 1329	<code>\um_map_char:nn</code> 613, 616,
<code>\um_config_mathbfsfit_Greek:</code>	... 1445		620–625, 628–633, 785, 884, 887,
<code>\um_config_mathbfsfit_greek:</code>	... 1450		911, 914, 920, 923, 929–934, 937–942
<code>\um_config_mathbfsfit_Latin:</code>	... 1438	<code>\um_map_char:nn</code> <u>766</u>

\um_map_chars_greek:nn	1046, 1050, 1051, 1055–1061,
... 612, 615, 619, 627, 779, 886,	1065, 1069–1075, 1081–1085,
888, 889, 896–899, 919, 922, 928, 936	1088–1092, 1096–1103, 1107–1110,
\um_map_chars_latin:nn	1114–1118, 1122, 1136, 1140,
602, 604, 607, 609, 776, 881–883,	1142, 1154, 1167, 1177,
885, 892–895, 903, 905, 910, 913	1194, 1198, 1200, 1205,
\um_map_chars_numbers:nn ... 589, 782	1209–1212, 1215, 1217, 1219,
\um_map_chars_range:nnn	1224, 1228–1241, 1245–1250,
..... 766, 777, 780, 783, 786	1252–1257, 1259, 1270, 1274,
\um_mathmap:Nnn 465, 472, 791, 798	1275, 1280–1286, 1296,
\um_mathmap_noparse:Nnn 465, <u>639</u>	1300–1302, 1306–1313, 1320,
\um_mathmap_parse:Nnn 472, <u>644</u>	1327, 1344, 1348, 1350,
\um_maybe_init_alphabet:n	1358–1361, 1364, 1366,
..... 467, 474, 950, 970	1368, 1376–1389, 1392,
\um_Nabla_up_or_it_usv .. 250, 254, 512	1394–1399, 1401–1406, 1418,
\um_nprimes:n 1494, 1504, 1507, 1510, 1513	1422, 1423, 1427–1433,
\um_nprimes_select:n 1500, 1531	1443, 1447, 1448, 1452–1458
\um_partial_up_or_it_usv 268, 272, 513	\um_set_mathalphabet_greek:Nnn ..
\um_peek_execute_branches_ss: 810, 1027, 1032, 1049,
..... 1663, 1668	1054, 1207, 1208, 1214, 1226,
\um_peek_execute_branches_ss_aux:	1227, 1243, 1273, 1279, 1299,
..... 1675, 1688	1305, 1356, 1357, 1363, 1374,
\um_prepare_alph:n 998, <u>1004</u>	1375, 1391, 1421, 1426, 1446, 1451
\um_process_symbol_noparse:nnnn .	\um_set_mathalphabet_latin:Nnn ..
..... 464, <u>492</u> 807, 1020,
\um_process_symbol_parse:nnnn 471, <u>492</u>	1023, 1042, 1045, 1064, 1068,
\um_remap_symbol:nnn	1095, 1106, 1113, 1121, 1126,
..... 466, 473, 501, 502, 504,	1127, 1129, 1134, 1135, 1138,
507–510, 512, 513, 516–523, 525–528	1153, 1157, 1163, 1166, 1173,
\um_remap_symbol_noparse:nnn 466, <u>500</u>	1176, 1184, 1185, 1187, 1192,
\um_remap_symbol_parse:nnn	1193, 1196, 1266, 1269, 1278,
..... 473, <u>500</u> , 531	1292, 1295, 1316, 1319, 1323,
\um_remap_symbols: 480, <u>500</u>	1326, 1334, 1335, 1337, 1342,
\um_scan_sscript: 1578, 1611, 1648, 1650	1343, 1346, 1414, 1417, 1439, 1442
\um_scan_sscript:TF 1649, 1658	\um_set_mathalphabet_numbers:Nnn
\um_scanprime:	804, 1078, 1147, 1150, 1160, 1170,
..... 1516, 1525, 1536, 1540, 1541	1180, 1263, 1289, 1330, 1411, 1436
\um_scanprime_collect:	\um_set_mathcode:nnnn <u>414</u> , 538, 655, 769
..... 1518, 1520, 1523, 1526, 1529	\um_setup_active_subscript:nn ...
\um_set_delcode:n ... 554, 557–579, 584 1602, 1616–1643
\um_set_delcode:nn 551–553, 555, 556, 581	\um_setup_active_superscript:nn .
\um_set_mathalph_range:Nnn <u>795</u> 1569, 1583–1599
\um_set_mathalph_range:nNnn	\um_setup_alphabets: 484, 945
..... 795, 805, 808, 811	\um_setup_alphanum: 483, <u>587</u>
\um_set_mathalphabet_char:Nnn 788,	\um_setup_bf_literals: 599, <u>891</u>
1024, 1028, 1029, 1033–1039,	\um_setup_bfshapes: 206, 490

