

Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/10/22 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			
2	Acknowledgements	3			
3	Getting started	3			
3.1	Package options	3			
4	Unicode maths font setup	4			
4.1	Using multiple fonts	5			
4.2	Script and scriptscript fonts/features	6			
5	Maths input	6			
5.1	Math ‘style’	6			
5.2	Bold style	7			
5.3	Sans serif style	8			
5.4	All (the rest) of the mathematical alphabets	9			
5.5	Miscellanea	10			
I	The unicode-math package	15			
6	Things we need	15			
6.1	Options	21			
6.2	Overcoming \@on-	26			
6.3	Other things	26			
7	Fundamentals	27			
7.1	Enlarging the number of maths families	27			
			7.2	\DeclareMathSymbol for unicode ranges	27
			7.3	The main \setmathfont macro	29
			7.4	(Big) operators	35
			7.5	Radicals	38
			7.6	Delimiters	39
			7.7	Maths accents	41
			8	Font features	42
			8.1	OpenType maths font features	42
			8.2	Script and scriptscript font options	43
			8.3	Range processing	43
			8.4	Resolving Greek symbol name control sequences	49
			9	Maths alphabets mapping definitions	50
			9.1	Alphabets	55
			10	Definitions of the math symbols	66
			11	Epilogue	67
			12	stix table data extraction	76
			A	Documenting maths support in the NFSS	77
			B	X_YTeX math font dimensions	79

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 $\text{Lua}\TeX$.

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 $\text{Xe}\TeX$; Barbara Beeton for her prodigious effort compiling the definitive list of unicode math glyphs and their $\text{L}\text{A}\text{T}\text{E}\text{X}$ 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 Salomon, and Khaled Hosny have been fantastic beta testers.

3 Getting started

Load unicode-math as a regular $\text{L}\text{A}\text{T}\text{E}\text{X}$ 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>script-font</code>	Font to use for sub- and super-scripts	section §4.2
<code>script-features</code>	Font features for sub- and super-scripts	section §4.2
<code>sscript-font</code>	Font to use for nested sub- and super-scripts	section §4.2
<code>sscript-features</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 `srix` 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 `\int`, or whole collection of symbols with the same math type, such as `\mathopen`, or complete math alphabets such as `\mathbb`. (Only numerical slots, however, can be used in ranged declarations.)

4.1.1 Control over maths alphabets

Exact control over maths alphabets can be somewhat involved. Here is the current plan.

- `[range=\mathbb]` to use the font for ‘bb’ letters only.
- `[range=\mathbfsfit/{greek,Greek}]` for Greek lowercase and uppercase only (with `latin`, `Latin`, `num` as well for Latin lower-/upper-case and numbers).
- `[range=\mathsf->\mathbfsfit]` to map to different output alphabet(s) (which is rather useless right now but will become less useless in the future).

And now the trick. If a particular math alphabet is not defined in the font, fall back onto the lower-base plane (i.e., upright) glyphs. Therefore, to use an ASCII-encoded fractur font, for example, write

```
\setmathfont[range=\mathfrak]{SomeFrakturFont}
```

and because the math plane fractur glyphs will be missing, unicode-math will know to use the ASCII ones instead. If necessary (but why?) this behaviour can be forced with `[range=\mathfrac->\mathup]`.

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.

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. Finally, it is not unknown to use upright letters for all characters, as seen in the Euler fonts.

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 four arguments: `TeX`, `ISO`, `French`, or `upright` (case insensitive).

The philosophy behind the interface to the mathematical alphabet symbols lies in L^ATeX'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.

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)	$(\alpha, \beta, \Gamma, \Xi)$
<code>math-style=upright</code>	(a, z, B, X)	$(\alpha, \beta, \Gamma, \Xi)$

Alternative interface However, some users may not like this convention of normalising their input. 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.

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 $\xi = (\xi_r, \xi_\varphi, \xi_\theta)$. Confusingly, the syntax in \LaTeX has been different for these two examples: `\mathbf{f}` in the former (‘ \mathbf{M} ’), and `\bm` (or `\boldsymbol`, deprecated) in the latter (‘ ξ ’).

In `unicode-math`, the `\mathbf{f}` 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=upright` 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.

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

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

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`, `\mathbfsup`, and `\mathbfsfit` 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=upright]` and `[sans-style=italic]` to control the behaviour of `\mathsf`. The `upright` style sets up the command to use the seemingly-useless upright sans serif, including Greek; the `italic` 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 `\mathbfsup` or `\mathbfsfit` based on `[sans-style=upright]` or `[sans-style=italic]`, 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 ’.

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`.

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

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 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: $\mathbb{D}, \mathbb{d}, \mathbb{E}, \mathbb{e}, \mathbb{J}$. These can be accessed (if not with their literal characters or control sequences) with the `\mathbbit`

Table 6: The various forms of nabla.

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

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.

The package 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`, `\primetripel`, and `\primequadruple`.

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

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.

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

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>

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

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

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

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

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

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

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

²§4.4.5.11 <http://www.w3.org/TR/2001/REC-MathML3/>

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 [2009/10/22 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
```

Extras we need to define:

```
7 \cs_generate_variant:Nn \tl_put_right:Nn {cx}
8 \cs_generate_variant:Nn \seq_if_in:NnTF {NV}
9 \cs_generate_variant:Nn \prop_gput:Nnn {Nxn}
10 \cs_generate_variant:Nn \prop_get:NnN {cxN}
11 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}
```

Counters and conditionals

```
12 \int_new:N \g_um_fam_int
13 \bool_new:N \l_um_fontspec_feature_bool
14 \bool_new:N \l_um_ot_math_bool
15 \bool_new:N \l_um_init_bool
```

For math-style:

```
16 \bool_new:N \g_um_literal_bool
17 \bool_new:N \g_um_upLatin_bool
18 \bool_new:N \g_um_uplatin_bool
19 \bool_new:N \g_um_upGreek_bool
20 \bool_new:N \g_um_upgreek_bool
```

For bold-style:

```
21 \bool_new:N \g_um_bfliteral_bool
22 \bool_new:N \g_um_bfupLatin_bool
23 \bool_new:N \g_um_bfuplatin_bool
24 \bool_new:N \g_um_bfupGreek_bool
25 \bool_new:N \g_um_bfupgreek_bool
```

For nabla:

```
26 \bool_new:N \g_um_upNabla_bool
27 \bool_new:N \g_um_uppartial_bool
28 \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

Rather than 'readable', in the end, this makes the code more extensible.

```
29 \cs_new:Nn \usv_set:nnn {
30   \tl_set:cn { \um_to_usv:nn {#1}{#2} } {#3}
31 }
32 \cs_new:Nn \um_to_usv:nn { g_um_#1_#2_usv }
33
34
35 \usv_set:nnn {up}{B}{`\B}
36 \usv_set:nnn {up}{C}{`\C}
37 \usv_set:nnn {up}{D}{`\D}
38 \usv_set:nnn {up}{E}{`\E}
39 \usv_set:nnn {up}{F}{`\F}
40 \usv_set:nnn {up}{H}{`\H}
41 \usv_set:nnn {up}{I}{`\I}
42 \usv_set:nnn {up}{L}{`\L}
43 \usv_set:nnn {up}{M}{`\M}
44 \usv_set:nnn {up}{N}{`\N}
45 \usv_set:nnn {up}{P}{`\P}
46 \usv_set:nnn {up}{Q}{`\Q}
47 \usv_set:nnn {up}{R}{`\R}
48 \usv_set:nnn {up}{Z}{`\Z}
49
```

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


```

50 \usv_set:nnn {it}{B}{"1D435}
51 \usv_set:nnn {it}{C}{"1D436}
52 \usv_set:nnn {it}{D}{"1D437}
53 \usv_set:nnn {it}{E}{"1D438}
54 \usv_set:nnn {it}{F}{"1D439}
55 \usv_set:nnn {it}{H}{"1D43B}
56 \usv_set:nnn {it}{I}{"1D43C}
57 \usv_set:nnn {it}{L}{"1D43F}
58 \usv_set:nnn {it}{M}{"1D440}
59 \usv_set:nnn {it}{N}{"1D441}
60 \usv_set:nnn {it}{P}{"1D443}
61 \usv_set:nnn {it}{Q}{"1D444}
62 \usv_set:nnn {it}{R}{"1D445}
63 \usv_set:nnn {it}{Z}{"1D44D}
64
65 \usv_set:nnn {up}{d}{`d}
66 \usv_set:nnn {up}{e}{`e}
67 \usv_set:nnn {up}{g}{`g}
68 \usv_set:nnn {up}{h}{"0068}
69 \usv_set:nnn {up}{i}{`i}
70 \usv_set:nnn {up}{j}{`j}
71 \usv_set:nnn {up}{o}{`o}
72
73 \usv_set:nnn {it}{d}{"1D451}
74 \usv_set:nnn {it}{e}{"1D452}
75 \usv_set:nnn {it}{g}{"1D454}
76 \usv_set:nnn {it}{h}{"210E}
77 \usv_set:nnn {it}{i}{"1D456}
78 \usv_set:nnn {it}{j}{"1D457}
79 \usv_set:nnn {it}{o}{"1D45C}
80
81 \usv_set:nnn {up}{num}{48}
82 \usv_set:nnn {up}{Latin}{65}
83 \usv_set:nnn {up}{latin}{97}
84 \usv_set:nnn {up}{Greek}{"391}
85 \usv_set:nnn {up}{greek}{"3B1}
86 \usv_set:nnn {it}{Latin}{"1D434}
87 \usv_set:nnn {it}{latin}{"1D44E}
88 \usv_set:nnn {it}{Greek}{"1D6E2}
89 \usv_set:nnn {it}{greek}{"1D6FC}
90 \usv_set:nnn {bb}{num}{"1D7D8}
91 \usv_set:nnn {bb}{Latin}{"1D538}
92 \usv_set:nnn {bb}{latin}{"1D552}
93 \usv_set:nnn {scr}{Latin}{"1D49C}
94 \usv_set:nnn {scr}{latin}{"1D4B6}
95 \usv_set:nnn {frak}{Latin}{"1D504}

```

```

96 \usv_set:nnn {frac}{latin}{ "1D51E}
97 \usv_set:nnn {sf}{num}{ "1D7E2}
98 \usv_set:nnn {sfup}{num}{ "1D7E2}
99 \usv_set:nnn {sfit}{num}{ "1D7E2}
100 \usv_set:nnn {sfup}{Latin}{ "1D5A0}
101 \usv_set:nnn {sf}{Latin}{ "1D5A0}
102 \usv_set:nnn {sfup}{latin}{ "1D5BA}
103 \usv_set:nnn {sf}{latin}{ "1D5BA}
104 \usv_set:nnn {sfit}{Latin}{ "1D608}
105 \usv_set:nnn {sfit}{latin}{ "1D622}
106 \usv_set:nnn {tt}{num}{ "1D7F6}
107 \usv_set:nnn {tt}{Latin}{ "1D670}
108 \usv_set:nnn {tt}{latin}{ "1D68A}

```

Bold:

```

109 \usv_set:nnn {bf}{num}{ "1D7CE}
110 \usv_set:nnn {bfup}{num}{ "1D7CE}
111 \usv_set:nnn {bfit}{num}{ "1D7CE}
112 \usv_set:nnn {bfup}{Latin}{ "1D400}
113 \usv_set:nnn {bfup}{latin}{ "1D41A}
114 \usv_set:nnn {bfup}{Greek}{ "1D6A8}
115 \usv_set:nnn {bfup}{greek}{ "1D6C2}
116 \usv_set:nnn {bfit}{Latin}{ "1D468}
117 \usv_set:nnn {bfit}{latin}{ "1D482}
118 \usv_set:nnn {bfit}{Greek}{ "1D71C}
119 \usv_set:nnn {bfit}{greek}{ "1D736}
120 \usv_set:nnn {bffrak}{Latin}{ "1D56C}
121 \usv_set:nnn {bffrak}{latin}{ "1D586}
122 \usv_set:nnn {bfscr}{Latin}{ "1D4D0}
123 \usv_set:nnn {bfscr}{latin}{ "1D4EA}
124 \usv_set:nnn {bfsf}{num}{ "1D7EC}
125 \usv_set:nnn {bfsfup}{num}{ "1D7EC}
126 \usv_set:nnn {bfsfit}{num}{ "1D7EC}
127 \usv_set:nnn {bfsfup}{Latin}{ "1D5D4}
128 \usv_set:nnn {bfsfup}{latin}{ "1D5EE}
129 \usv_set:nnn {bfsfup}{Greek}{ "1D756}
130 \usv_set:nnn {bfsfup}{greek}{ "1D770}
131 \usv_set:nnn {bfsfit}{Latin}{ "1D63C}
132 \usv_set:nnn {bfsfit}{latin}{ "1D656}
133 \usv_set:nnn {bfsfit}{Greek}{ "1D790}
134 \usv_set:nnn {bfsfit}{greek}{ "1D7AA}
135 \usv_set:nnn {bfsf}{Latin}{ \bool_if:NTF \g_um_upLatin_bool \g_um_bfsfup_Latin_usv \g_um_bfsf
136 \usv_set:nnn {bfsf}{latin}{ \bool_if:NTF \g_um_uplatin_bool \g_um_bfsfup_latin_usv \g_um_bfsf
137 \usv_set:nnn {bfsf}{Greek}{ \bool_if:NTF \g_um_upGreek_bool \g_um_bfsfup_Greek_usv \g_um_bfsf
138 \usv_set:nnn {bfsf}{greek}{ \bool_if:NTF \g_um_upgreek_bool \g_um_bfsfup_greek_usv \g_um_bfsf
139 \usv_set:nnn {bf}{Latin}{ \bool_if:NTF \g_um_bfupLatin_bool \g_um_bfup_Latin_usv \g_um_bfit_L
140 \usv_set:nnn {bf}{latin}{ \bool_if:NTF \g_um_bfuplatin_bool \g_um_bfup_latin_usv \g_um_bfit_L

```

141 \usv_set:nnn {bf}{Greek}{ \bool_if:NTF \g_um_bfupGreek_bool \g_um_bfup_Greek_usv \g_um_bfit_G
 142 \usv_set:nnn {bf}{greek}{ \bool_if:NTF \g_um_bfupgreek_bool \g_um_bfup_greek_usv \g_um_bfit_g

Greek variants:

143 \usv_set:nnn {up}{varTheta}{ "3F4}
 144 \usv_set:nnn {up}{Digamma}{ "3DC}
 145 \usv_set:nnn {up}{varepsilon}{ "3F5}
 146 \usv_set:nnn {up}{vartheta}{ "3D1}
 147 \usv_set:nnn {up}{varkappa}{ "3F0}
 148 \usv_set:nnn {up}{varphi}{ "3D5}
 149 \usv_set:nnn {up}{varrho}{ "3F1}
 150 \usv_set:nnn {up}{varpi}{ "3D6}
 151 \usv_set:nnn {up}{digamma}{ "3DD}

Bold:

152 \usv_set:nnn {bfup}{varTheta}{ "1D6B9}
 153 \usv_set:nnn {bfup}{Digamma}{ "1D7CA}
 154 \usv_set:nnn {bfup}{varepsilon}{ "1D6DC}
 155 \usv_set:nnn {bfup}{vartheta}{ "1D6DD}
 156 \usv_set:nnn {bfup}{varkappa}{ "1D6DE}
 157 \usv_set:nnn {bfup}{varphi}{ "1D6DF}
 158 \usv_set:nnn {bfup}{varrho}{ "1D6E0}
 159 \usv_set:nnn {bfup}{varpi}{ "1D6E1}
 160 \usv_set:nnn {bfup}{digamma}{ "1D7CB}

Italic Greek variants:

161 \usv_set:nnn {it}{varTheta}{ "1D6F3}
 162 \usv_set:nnn {it}{varepsilon}{ "1D716}
 163 \usv_set:nnn {it}{vartheta}{ "1D717}
 164 \usv_set:nnn {it}{varkappa}{ "1D718}
 165 \usv_set:nnn {it}{varphi}{ "1D719}
 166 \usv_set:nnn {it}{varrho}{ "1D71A}
 167 \usv_set:nnn {it}{varpi}{ "1D71B}

Bold italic:

168 \usv_set:nnn {bfit}{varTheta}{ "1D72D}
 169 \usv_set:nnn {bfit}{varepsilon}{ "1D750}
 170 \usv_set:nnn {bfit}{vartheta}{ "1D751}
 171 \usv_set:nnn {bfit}{varkappa}{ "1D752}
 172 \usv_set:nnn {bfit}{varphi}{ "1D753}
 173 \usv_set:nnn {bfit}{varrho}{ "1D754}
 174 \usv_set:nnn {bfit}{varpi}{ "1D755}

Bold sans:

175 \usv_set:nnn {bfsfup}{varTheta}{ "1D767}
 176 \usv_set:nnn {bfsfup}{varepsilon}{ "1D78A}
 177 \usv_set:nnn {bfsfup}{vartheta}{ "1D78B}
 178 \usv_set:nnn {bfsfup}{varkappa}{ "1D78C}
 179 \usv_set:nnn {bfsfup}{varphi}{ "1D78D}

180 \usv_set:nnn {bfsfup}{varrho}{1D78E}
 181 \usv_set:nnn {bfsfup}{varpi}{1D78F}

Bold sans italic:

182 \usv_set:nnn {bfsfit}{varTheta}{1D7A1}
 183 \usv_set:nnn {bfsfit}{varepsilon}{1D7C4}
 184 \usv_set:nnn {bfsfit}{vartheta}{1D7C5}
 185 \usv_set:nnn {bfsfit}{varkappa}{1D7C6}
 186 \usv_set:nnn {bfsfit}{varphi}{1D7C7}
 187 \usv_set:nnn {bfsfit}{varrho}{1D7C8}
 188 \usv_set:nnn {bfsfit}{varpi}{1D7C9}

Nabla:

189 \usv_set:nnn {up}{Nabla}{2207}
 190 \usv_set:nnn {it}{Nabla}{1D6FB}
 191 \usv_set:nnn {bfup}{Nabla}{1D6C1}
 192 \usv_set:nnn {bfit}{Nabla}{1D735}
 193 \usv_set:nnn {bfsfup}{Nabla}{1D76F}
 194 \usv_set:nnn {bfsfit}{Nabla}{1D7A9}

Partial:

195 \usv_set:nnn {up}{partial}{2202}
 196 \usv_set:nnn {it}{partial}{1D715}
 197 \usv_set:nnn {bfup}{partial}{1D6DB}
 198 \usv_set:nnn {bfit}{partial}{1D74F}
 199 \usv_set:nnn {bfsfup}{partial}{1D789}
 200 \usv_set:nnn {bfsfit}{partial}{1D7C3}

Latin 'h':

201 \usv_set:nnn {bb}{h}{1D559}
 202 \usv_set:nnn {tt}{h}{1D691}
 203 \usv_set:nnn {scr}{h}{1D4BD}
 204 \usv_set:nnn {frak}{h}{1D525}
 205 \usv_set:nnn {bfup}{h}{1D421}
 206 \usv_set:nnn {bfit}{h}{1D489}
 207 \usv_set:nnn {sfup}{h}{1D5C1}
 208 \usv_set:nnn {sfit}{h}{1D629}
 209 \usv_set:nnn {bffrak}{h}{1D58D}
 210 \usv_set:nnn {bfscr}{h}{1D4F1}
 211 \usv_set:nnn {bfsfup}{h}{1D5F5}
 212 \usv_set:nnn {bfsfit}{h}{1D65D}

Blackboard:

213 \usv_set:nnn {bb}{C}{2102}
 214 \usv_set:nnn {bb}{H}{210D}
 215 \usv_set:nnn {bb}{N}{2115}
 216 \usv_set:nnn {bb}{P}{2119}
 217 \usv_set:nnn {bb}{Q}{211A}
 218 \usv_set:nnn {bb}{R}{211D}

```

219 \usv_set:nnn {bb}{Z}{ "2124}
220 \usv_set:nnn {up}{Pi}      {"03A0}
221 \usv_set:nnn {up}{pi}      {"03C0}
222 \usv_set:nnn {up}{Gamma}    {"0393}
223 \usv_set:nnn {up}{gamma}    {"03B3}
224 \usv_set:nnn {up}{summation}{ "2211}
225 \usv_set:nnn {it}{Pi}      {"1D6F1}
226 \usv_set:nnn {it}{pi}      {"1D70B}
227 \usv_set:nnn {it}{Gamma}    {"1D6E4}
228 \usv_set:nnn {it}{gamma}    {"1D6FE}
229 \usv_set:nnn {bb}{Pi}      {"213F}
230 \usv_set:nnn {bb}{pi}      {"213C}
231 \usv_set:nnn {bb}{Gamma}    {"213E}
232 \usv_set:nnn {bb}{gamma}    {"213D}
233 \usv_set:nnn {bb}{summation}{ "2140}

```

Italic blackboard:

```

234 \usv_set:nnn {bbi}{D}{ "2145}
235 \usv_set:nnn {bbi}{d}{ "2146}
236 \usv_set:nnn {bbi}{e}{ "2147}
237 \usv_set:nnn {bbi}{i}{ "2148}
238 \usv_set:nnn {bbi}{j}{ "2149}

```

Script exceptions:

```

239 \usv_set:nnn {scr}{B}{ "212C}
240 \usv_set:nnn {scr}{E}{ "2130}
241 \usv_set:nnn {scr}{F}{ "2131}
242 \usv_set:nnn {scr}{H}{ "210B}
243 \usv_set:nnn {scr}{I}{ "2110}
244 \usv_set:nnn {scr}{L}{ "2112}
245 \usv_set:nnn {scr}{M}{ "2133}
246 \usv_set:nnn {scr}{R}{ "211B}
247 \usv_set:nnn {scr}{e}{ "212F}
248 \usv_set:nnn {scr}{g}{ "210A}
249 \usv_set:nnn {scr}{o}{ "2134}

```

Fraktur exceptions:

```

250 \usv_set:nnn {frak}{C}{ "212D}
251 \usv_set:nnn {frak}{H}{ "210C}
252 \usv_set:nnn {frak}{I}{ "2111}
253 \usv_set:nnn {frak}{R}{ "211C}
254 \usv_set:nnn {frak}{Z}{ "2128}

```

6.1 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.

```

255 \DeclareDocumentCommand \unimathsetup {m} {
256   \setkeys{unicode-math.sty}{#1}
257 }

```

math-style

```

258 \define@choicekey*{unicode-math.sty}
259   {math-style}[\@tempa\@tempb]{iso,tex,french,upright,literal}{
260   \ifcase\@tempb\relax
261     \bool_set_false:N \g_um_upGreek_bool
262     \bool_set_false:N \g_um_upgreek_bool
263     \bool_set_false:N \g_um_upLatin_bool
264     \bool_set_false:N \g_um_uplatin_bool
265     \bool_set_false:N \g_um_bfupGreek_bool
266     \bool_set_false:N \g_um_bfupgreek_bool
267     \bool_set_false:N \g_um_bfupLatin_bool
268     \bool_set_false:N \g_um_bfuplatin_bool
269     \bool_set_false:N \g_um_upNabla_bool
270     \bool_set_false:N \g_um_uppartial_bool
271     \bool_set_false:N \g_um_upsans_bool
272     \bool_set_false:N \g_um_texgreek_bool
273     \bool_set_false:N \g_um_literal_bool
274   \or
275     \bool_set_true:N \g_um_upGreek_bool
276     \bool_set_false:N \g_um_upgreek_bool
277     \bool_set_false:N \g_um_upLatin_bool
278     \bool_set_false:N \g_um_uplatin_bool
279     \bool_set_true:N \g_um_bfupGreek_bool
280     \bool_set_false:N \g_um_bfupgreek_bool
281     \bool_set_true:N \g_um_bfupLatin_bool
282     \bool_set_true:N \g_um_bfuplatin_bool
283     \bool_set_true:N \g_um_upNabla_bool
284     \bool_set_false:N \g_um_uppartial_bool
285     \bool_set_true:N \g_um_upsans_bool
286     \bool_set_false:N \g_um_texgreek_bool
287     \bool_set_false:N \g_um_literal_bool
288   \or
289     \bool_set_true:N \g_um_upGreek_bool
290     \bool_set_true:N \g_um_upgreek_bool
291     \bool_set_true:N \g_um_upLatin_bool
292     \bool_set_false:N \g_um_uplatin_bool
293     \bool_set_true:N \g_um_bfupGreek_bool
294     \bool_set_true:N \g_um_bfupgreek_bool
295     \bool_set_true:N \g_um_bfupLatin_bool
296     \bool_set_true:N \g_um_bfuplatin_bool

```

```

297 \bool_set_true:N \g_um_upNabla_bool
298 \bool_set_true:N \g_um_uppartial_bool
299 \bool_set_true:N \g_um_upsans_bool
300 \bool_set_false:N \g_um_texgreek_bool
301 \bool_set_false:N \g_um_literal_bool
302 \or
303 \bool_set_true:N \g_um_upGreek_bool
304 \bool_set_true:N \g_um_upgreek_bool
305 \bool_set_true:N \g_um_upLatin_bool
306 \bool_set_true:N \g_um_uplatin_bool
307 \bool_set_true:N \g_um_bfupGreek_bool
308 \bool_set_true:N \g_um_bfupgreek_bool
309 \bool_set_true:N \g_um_bfupLatin_bool
310 \bool_set_true:N \g_um_bfuplatin_bool
311 \bool_set_true:N \g_um_upNabla_bool
312 \bool_set_true:N \g_um_uppartial_bool
313 \bool_set_true:N \g_um_upsans_bool
314 \bool_set_false:N \g_um_texgreek_bool
315 \bool_set_false:N \g_um_literal_bool
316 \or
317 \bool_set_true:N \g_um_literal_bool
318 \bool_set_true:N \g_um_bfliteral_bool
319 \bool_set_true:N \g_um_sfliteral_bool
320 \bool_set_false:N \g_um_texgreek_bool
321 \fi
322 }

```

bold-style

```

323 \define@choicekey*{unicode-math.sty}{bold-style}[\@tempa\@tempb]{iso,tex,upright,literal}{
324 \ifcase\@tempb\relax
325 \bool_set_false:N \g_um_bfliteral_bool
326 \bool_set_false:N \g_um_bfupGreek_bool
327 \bool_set_false:N \g_um_bfupgreek_bool
328 \bool_set_false:N \g_um_bfupLatin_bool
329 \bool_set_false:N \g_um_bfuplatin_bool
330 \or
331 \bool_set_false:N \g_um_bfliteral_bool
332 \bool_set_true:N \g_um_bfupGreek_bool
333 \bool_set_false:N \g_um_bfupgreek_bool
334 \bool_set_true:N \g_um_bfupLatin_bool
335 \bool_set_true:N \g_um_bfuplatin_bool
336 \or
337 \bool_set_false:N \g_um_bfliteral_bool
338 \bool_set_true:N \g_um_bfupGreek_bool
339 \bool_set_true:N \g_um_bfupgreek_bool
340 \bool_set_true:N \g_um_bfupLatin_bool

```

```

341 \bool_set_true:N \g_um_bfuplatin_bool
342 \or
343 \bool_set_true:N \g_um_bfliteral_bool
344 \fi
345 }

```

sans-style

```

346 \bool_new:N \g_um_upsans_bool
347 \bool_new:N \g_um_sfliteral_bool
348 \define@choicekey*{unicode-math.sty}{
349   {sans-style}[\@tempa\@tempb]{italic,upright,literal}{
350   \ifcase\@tempb\relax
351     \bool_set_false:N \g_um_upsans_bool
352   \or
353     \bool_set_true:N \g_um_upsans_bool
354   \or
355     \bool_set_true:N \g_um_sfliteral_bool
356   \fi
357 }

```

Symbol obliqueness

```

358 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
359   \ifcase\@tempb
360     \bool_set_true:N \g_um_upNabla_bool
361   \or
362     \bool_set_false:N \g_um_upNabla_bool
363   \fi
364 }
365 \cs_set:Nn \um_setup_nabla: {
366   \bool_if:NTF \g_um_upNabla_bool {
367     \tl_set:Nn \g_um_Nabla_up_or_it_usv { \g_um_up_Nabla_usv }
368     \tl_set:Nn \g_um_bfNabla_up_or_it_usv { \g_um_bfup_Nabla_usv }
369     \tl_set:Nn \g_um_bfsfNabla_up_or_it_usv { \g_um_bfsfup_Nabla_usv }
370   }{
371     \tl_set:Nn \g_um_Nabla_up_or_it_usv { \g_um_it_Nabla_usv }
372     \tl_set:Nn \g_um_bfNabla_up_or_it_usv { \g_um_bfit_Nabla_usv }
373     \tl_set:Nn \g_um_bfsfNabla_up_or_it_usv { \g_um_bfsfit_Nabla_usv }
374   }
375 }
376 \define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
377   \ifcase\@tempb
378     \bool_set_true:N \g_um_uppartial_bool
379   \or
380     \bool_set_false:N \g_um_uppartial_bool
381   \fi

```



```

382 }
383 \cs_set:Nn \um_setup_partial: {
384   \bool_if:NTF \g_um_uppartial_bool {
385     \tl_set:Nn \g_um_partial_up_or_it_usv { \g_um_up_partial_usv }
386     \tl_set:Nn \g_um_bfpartial_up_or_it_usv { \g_um_bfup_partial_usv }
387     \tl_set:Nn \g_um_bfsfpartial_up_or_it_usv { \g_um_bfsfup_partial_usv }
388   }{
389     \tl_set:Nn \g_um_partial_up_or_it_usv { \g_um_it_partial_usv }
390     \tl_set:Nn \g_um_bfpartial_up_or_it_usv { \g_um_bfit_partial_usv }
391     \tl_set:Nn \g_um_bfsfpartial_up_or_it_usv { \g_um_bfsfit_partial_usv }
392   }
393 }

```

Epsilon and phi shapes

```

394 \define@choicekey*{unicode-math.sty}{vargreek-shape}[\@tempa\@tempb]{unicode,TeX}{
395   \ifcase\@tempb
396     \bool_set_false:N \g_um_texgreek_bool
397   \or
398     \bool_set_true:N \g_um_texgreek_bool
399   \fi
400 }

```

Colon style

```

401 \bool_new:N \g_um_literal_colon_bool
402 \define@choicekey*{unicode-math.sty}{colon}[\@tempa\@tempb]{literal,TeX}{
403   \ifcase\@tempb
404     \bool_set_true:N \g_um_literal_colon_bool
405   \or
406     \bool_set_false:N \g_um_literal_colon_bool
407   \fi
408 }

```

Slash delimiter style

```

409 \define@choicekey*{unicode-math.sty}{slash-delimiter}[\@tempa\@tempb]{ascii,frac,div}{
410   \ifcase\@tempb
411     \tl_set:Nn \g_um_slash_delimiter_usv {"002F}
412   \or
413     \tl_set:Nn \g_um_slash_delimiter_usv {"2044}
414   \or
415     \tl_set:Nn \g_um_slash_delimiter_usv {"2215}
416   \fi
417 }

418 \ExecuteOptionsX{math-style=TeX,slash-delimiter=ascii}
419 \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.

```

420 \tl_map_inline:nn {
421   \new@mathgroup\cdp@list\cdp@elt\DeclareMathSizes
422   \@DeclareMathSizes\newmathalphabet\newmathalphabet@@\newmathalphabet@@@
423   \DeclareMathVersion\define@mathalphabet\define@mathgroup\addtoversion
424   \version@list\version@elt\alpha@list\alpha@elt
425   \restore@mathversion\init@restore@version\dorestore@version\process@table
426   \new@mathversion\DeclareSymbolFont\group@list\group@elt
427   \new@symbolfont\SetSymbolFont\SetSymbolFont@\get@cdp
428   \DeclareMathAlphabet\new@mathalphabet\SetMathAlphabet\SetMathAlphabet@
429   \DeclareMathAccent\set@mathaccent\DeclareMathSymbol\set@mathchar
430   \set@mathsymbol\DeclareMathDelimiter\@xxDeclareMathDelimiter\@DeclareMathDelimiter
431   \@xxDeclareMathDelimiter\set@mathdelimiter\set@@mathdelimiter\DeclareMathRadical
432   \mathchar@type\DeclareSymbolFontAlphabet\DeclareSymbolFontAlphabet@
433 }{
434   \tl_remove_in:Nn \@preamblecmds {\do#1}
435 }

```

6.3 Other things

`\um_fontdimen_to_percent:nn` **#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.

```

436 \def\um_fontdimen_to_percent:nn#1#2{
437   0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax
438 }

```

`\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.

```

439 \def\um@scaled@apply#1#2#3{
440   \ifx#1\scriptstyle
441     #2\um_fontdimen_to_percent:nn{10}\l_um_font#3
442   \else
443     \ifx#1\scriptscriptstyle
444       #2\um_fontdimen_to_percent:nn{11}\l_um_font#3
445     \else
446       #2#3%
447     \fi

```

```

448 \fi
449 }

```

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 `ltxssbas.dtx`) we want to redefine

```

450 \def\new@mathgroup{\alloc@8\mathgroup\chardef\@cclvi}
451 \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`.

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

```

\um_set_mathsymbol:nNNn #1 : Symbol font number, e.g., \symoperators
                        #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.

```

452 \cs_set:Nn \um_set_mathsymbol:nNNn {

```

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

```

453 \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 `\sum_sym`.

```

454 \begingroup
455 \char_make_active:n {#4}
456 \global\mathcode#4="8000\relax
457 \um@scanactivedef #4 \@nil { \csname\cs_to_str:N #2 _sym\endcsname }
458 \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 `\sumop`.

```
459 \expandafter\global\expandafter\XeTeXmathchardef
460 \csname\cs_to_str:N #2 op\endcsname =" \mathchar@type#3 #1 #4\relax
```

Now define `\sum_sym` as `\sumop`, followed by `\nolimits` if necessary.

```
461 \cs_gset:cpx { \cs_to_str:N #2 _sym } {
462   \exp_not:c { \cs_to_str:N #2 op}
463   \exp_not:n { \tl_if_in:NnT \l_um_nolimits_tl {#2} \nolimits}
464 }
```

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

```
465 \else
```

Delimiters and radicals Sqrt radical is defined as a csmathopen.

```
466 \ifx\mathopen#3\relax
467   \tl_if_in:NnTF \l_um_radicals_tl #2 {
468     \cs_gset:cpn { \cs_to_str:N #2 sign} { \XeTeXradical #1 #4 \relax }
469   }{
470     \cs_gset:Npn #2 { \XeTeXdelimiter " \mathchar@type#3 #1 #4 \relax}
471     \global\XeTeXdelcode#4=#1 #4 \relax
472     \global\XeTeXmathcode#4=" \mathchar@type#3 #1 #4 \relax
473   }
474 \else
475   \ifx\mathclose#3\relax
476     \cs_gset:Npn #2 { \XeTeXdelimiter " \mathchar@type#3 #1 #4 \relax}
477     \global\XeTeXdelcode#4=#1 #4 \relax
478     \global\XeTeXmathcode#4=" \mathchar@type#3 #1 #4 \relax
479   \else
```

Fences

```
480 \ifx\mathfence#3
481   \global\XeTeXmathcode#4=" \mathchar@type#3 #1 #4 \relax
482   \global\XeTeXdelcode#4=#1 #4 \relax
483   \cs_gset:cpn {l \cs_to_str:N #2} { \XeTeXdelimiter " \math-
484     char@type\mathopen #1 #4 \relax}
485   \cs_gset:cpn {r \cs_to_str:N #2} { \XeTeXdelimiter " \math-
486     char@type\mathclose #1 #4 \relax}
487 \else
```

Accents

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

And finally, the general case. We define the unicode mathcode for the character. The macro is defined later on generically in terms of the unicode character.

```

489         \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
490     \fi
491 \fi
492 \fi
493 \fi
494 \fi
495 }

```

`\um_set_mathcode:nnnn` Note that this declaration *isn't* global so that it can be constrained by grouping inside math alphabet switches.

```

496 \cs_set:Nn \um_set_mathcode:nnnn {
497     \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
498 }

```

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

```

499 \DeclareDocumentCommand \setmathfont { O{ } m } {

```

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

```

500     \let\glb@currsizel\relax

```

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

```

501     \bool_set_true:N \l_um_init_bool
502     \seq_clear:N \l_um_char_range_seq
503     \let\um@char@num@range\@empty

```

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

```

504     \csname S@\f@size\endcsname

```

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

```

505 \tl_set:Nn \l_um_mversion_tf {normal}
506 \DeclareMathVersion{\l_um_mversion_tf}

```

Define default font features for the script and scriptscript font.

```

507 \tl_set:Nn \l_um_script_features_tl {ScriptStyle}
508 \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
509 \tl_set:Nn \l_um_script_font_tl      {#2}
510 \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.

```

511 \setkeys*{unicode-math.sty}{#1}
512 \cs_set:Npx \um_tmp: {
513   \exp_not:N \setkeys*[um]{options}{\exp_not:V \XKV@rm}
514 }
515 \um_tmp:
516 \cs_set:Npx \um_tmp: {
517   \exp_not:N \zf@fontspec {
518     BoldFont = {}, ItalicFont = {},
519     Script = Math,
520     SizeFeatures = {
521       {Size = \tf@size-} ,
522       {Size = \sf@size-\tf@size ,
523         Font = \l_um_script_font_tl ,
524         \l_um_script_features_tl
525       } ,
526       {Size = -\sf@size ,
527         Font = \l_um_sscript_font_tl ,
528         \l_um_sscript_features_tl
529       }
530     },
531     \XKV@rm
532   }{#2}
533 }
534 \bool_set_true:N \l_um_fontspec_feature_bool
535 \um_tmp:
536 \bool_set_false:N \l_um_fontspec_feature_bool

```

Check for the correct number of `\fontdimens`:

```

537 \font\l_um_font="#2"\relax
538 %% \ifdim \dimexpr\fontdimen9\l_um_font*65536\relax =65pt\relax
539 %% \bool_set_true:N \l_um_ot_math_bool
540 %% \else
541 %% \bool_set_false:N \l_um_ot_math_bool
542 %% \PackageWarningNoLine{unicode-math}{
543 %%   The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
544 %%   Some~ maths~ features~ will~ not~ be~ available~ or~ behave~

```

```

545 %%      in~ a~ substandard~ manner
546 %%    }
547 %%  \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

```

548 \bool_if:NTF \l_um_init_bool {
549   \tl_set:Nn \um_symfont_tl {um_allsym}
550   \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
551   \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
552   \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
553   \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
554   \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
555   \cs_set_eq:NN \um_map_char_internal:nn \um_map_char_noparse:nn
556 }{
557   \int_incr:N \g_um_fam_int
558   \tl_set:Nx \um_symfont_tl {um_fam\int_use:N\g_um_fam_int}
559   \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn
560   \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
561   \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
562   \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
563   \cs_set_eq:NN \um_map_char_internal:nn \um_map_char_parse:nn
564 }

```

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

```

565 \DeclareSymbolFont{\um_symfont_tl}
566   {\encodingdefault}{\zf@family}{\mddefault}{\updefault}

```

And now we input every single maths char. See File 12 for the source to `unicode-math.tex` which is used to create `unicode-math-table.tex`.

```

567 \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
- Assign delimiter codes for symbols that need to grow
- Setup the maths alphabets (`\mathbf` etc.)

```

568 \um_setup_nabla:
569 \um_setup_partial:
570 \um_remap_symbols:
571 \um_setup_mathactives:
572 \um_setup_delcodes:
573 \um_setup_alphabets:
574 }

```

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
`\um_process_symbol_parse:nnnn` are to be defined. See section §8.3 for the code that enables this.

```

575 \cs_set:Nn \um_process_symbol_noparse:nnnn {
576   \exp_args:Nc \um_set_mathsymbol:nNNn {sym\um_symfont_tl}#2#3{#1}
577 }

578 \cs_set:Nn \um_process_symbol_parse:nnnn {
579   \um@parse@term{#1}{#2}{#3}{
580     \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
581   }
582 }

```

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

```

583 \cs_new:Nn \um_remap_symbols: {
584   \um_remap_symbol:nnn{\-}{\mathbin}{"02212}% hyphen to minus
585   \um_remap_symbol:nnn{\*}{\mathbin}{"02217}% text asterisk to "cen-
      tred asterisk"
586   \bool_if:NF \g_um_literal_colon_bool {
587     \um_remap_symbol:nnn{\:}{\mathrel}{"02236}% colon to ratio (i.e., punct to rel)
588   }
589   \bool_if:NTF \g_um_literal_bool {
590     \um_remap_symbol:nnn {\g_um_up_Nabla_usv}{\mathord}{\g_um_up_Nabla_usv}
591     \um_remap_symbol:nnn {\g_um_it_Nabla_usv}{\mathord}{\g_um_it_Nabla_usv}
592     \um_remap_symbol:nnn {\g_um_up_partial_usv}{\mathord}{\g_um_up_partial_usv}
593     \um_remap_symbol:nnn {\g_um_it_partial_usv}{\mathord}{\g_um_it_partial_usv}
594   }{
595     \um_remap_symbol:nnn {\g_um_up_Nabla_usv,\g_um_it_Nabla_usv}{\mathord}{\g_um_Nabla_up_or_
596     \um_remap_symbol:nnn {\g_um_up_partial_usv,\g_um_it_partial_usv}{\mathord}{\g_um_partial_
597   }

```

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

```

598 \bool_if:NTF \g_um_bfliteral_bool {
599   \um_remap_symbol:nnn {\g_um_bfup_Nabla_usv }{\mathord}{\g_um_bfup_Nabla_usv}
600   \um_remap_symbol:nnn {\g_um_bfit_Nabla_usv }{\mathord}{\g_um_bfit_Nabla_usv}
601   \um_remap_symbol:nnn {\g_um_bfsfup_Nabla_usv }{\mathord}{\g_um_bfsfup_Nabla_usv}
602   \um_remap_symbol:nnn {\g_um_bfsfit_Nabla_usv }{\mathord}{\g_um_bfsfit_Nabla_usv}

```



```

603 \um_remap_symbol:nnn {\g_um_bfup_partial_usv }{\mathord}{\g_um_bfup_partial_usv}
604 \um_remap_symbol:nnn {\g_um_bfit_partial_usv }{\mathord}{\g_um_bfit_partial_usv}
605 \um_remap_symbol:nnn {\g_um_bfsfup_partial_usv}{\mathord}{\g_um_bfsfup_partial_usv}
606 \um_remap_symbol:nnn {\g_um_bfsfit_partial_usv}{\mathord}{\g_um_bfsfit_partial_usv}
607 }{
608 \um_remap_symbol:nnn {\g_um_bfup_Nabla_usv,\g_um_bfit_Nabla_usv}{\mathord}{\g_um_bfNabla_
609 \um_remap_symbol:nnn {\g_um_bfsfup_Nabla_usv,\g_um_bfsfit_Nabla_usv}{\mathord}{\g_um_bfsf
610 \um_remap_symbol:nnn {\g_um_bfup_partial_usv,\g_um_bfit_partial_usv}{\mathord}{\g_um_bfpa
611 \um_remap_symbol:nnn {\g_um_bfsfup_partial_usv,\g_um_bfsfit_partial_usv}{\mathord}{\g_um_
612 }
613 }

```

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

```

614 \cs_new:Nn \um_remap_symbol_parse:nnn {
615   \um@parse@term {#3} {\@nil} {#2} {
616     \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
617   }
618 }
619 \cs_new:Nn \um_remap_symbol_noparse:nnn {
620   \clist_map_inline:nn {#1} {
621     \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
622   }
623 }

```

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:`

```

624 \cs_new:Nn \um_setup_mathactives: {
625   \um_make_mathactive:nNN {"2032} \sprime \mathord
626 }

```

`\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!

```

627 \cs_new:Nn \um_make_mathactive:nNN {
628   \XeTeXmathchardef #2 = "\mathchar@type #3
629   \csname sym\um_symfont_tl\endcsname
630   #1 \scan_stop:
631   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:
632 }

```

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. And some of them are probably not meant to stretch, either. But adding them here doesn't hurt.

`\um_setup_delcodes:`

```

633 \cs_new:Nn \um_setup_delcodes: {
634   \um_set_delcode:nn {\ /} {\g_um_slash_delimiter_usv}
635   \um_set_delcode:nn {"2044} {\g_um_slash_delimiter_usv} % fracslash
636   \um_set_delcode:nn {"2215} {\g_um_slash_delimiter_usv} % divslash
637   \um_set_delcode:n {"005C} % backslash
638   \um_set_delcode:nn {\<} {"27E8} % angle brackets with ascii notation
639   \um_set_delcode:nn {\>} {"27E9} % angle brackets with ascii notation
640   \um_set_delcode:n {"2191} % up arrow
641   \um_set_delcode:n {"2193} % down arrow
642   \um_set_delcode:n {"2195} % updown arrow
643   \um_set_delcode:n {"219F} % up arrow twohead
644   \um_set_delcode:n {"21A1} % down arrow twohead
645   \um_set_delcode:n {"21A5} % up arrow from bar
646   \um_set_delcode:n {"21A7} % down arrow from bar
647   \um_set_delcode:n {"21A8} % updown arrow from bar
648   \um_set_delcode:n {"21BE} % up harpoon right
649   \um_set_delcode:n {"21BF} % up harpoon left
650   \um_set_delcode:n {"21C2} % down harpoon right
651   \um_set_delcode:n {"21C3} % down harpoon left
652   \um_set_delcode:n {"21C5} % arrows up down
653   \um_set_delcode:n {"21F5} % arrows down up
654   \um_set_delcode:n {"21C8} % arrows up up
655   \um_set_delcode:n {"21CA} % arrows down down
656   \um_set_delcode:n {"21D1} % double up arrow
657   \um_set_delcode:n {"21D3} % double down arrow
658   \um_set_delcode:n {"21D5} % double updown arrow
659   \um_set_delcode:n {"21DE} % up arrow double stroke
660   \um_set_delcode:n {"21DF} % down arrow double stroke
661   \um_set_delcode:n {"21E1} % up arrow dashed
662   \um_set_delcode:n {"21E3} % down arrow dashed
663   \um_set_delcode:n {"21E7} % up white arrow
664   \um_set_delcode:n {"21E9} % down white arrow
665   \um_set_delcode:n {"21EA} % up white arrow from bar
666   \um_set_delcode:n {"21F3} % updown white arrow
667 }
```

`\um_set_delcode:nn` : TODO : hook into range feature

```

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

```

670 }
671 \cs_new:Nn \um_set_delcode:n {
672   \XeTeXdelcode#1 = \csname sym\um_symfont_t1\endcsname #1
673 }

```

7.3.4 Maths alphabets' character mapping

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 def-
inition.

674 \cs_set:Nn \um_mathmap_noparse:Nnn {
675   \clist_map_inline:nn {#2} {
676     \tl_put_right:cx {um_setup\cs_to_str:N #1:} {
677       \exp_not:N\um_set_mathcode:nnnn{##1}{\exp_not:N\mathalpha}{\um_symfont_t1}{#3}
678     }
679   }
680 }

\um_mathmap_parse:Nnn #1 : Maths alphabet, e.g., \mathbb
#2 : Input slot(s), e.g., the slot for 'A' (comma separated)
#3 : Output slot, e.g., the slot for 'A'
When \um@parse@term is executed, it populates the \um@char@num@range macro
with slot numbers corresponding to the specified range. This range is used to
conditionally add \um_set_mathcode:nnnn declaractions to the maths alphabet
definition.

681 \cs_set:Nn \um_mathmap_parse:Nnn {
682   \clist_map_inline:Nn \um@char@num@range {
683     \ifnum##1=#3\relax
684       \um_mathmap_noparse:Nnn {#1}{#2}{#3}
685     \fi
686   }
687 }

```

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

Following is a table of every math operator (`\mathop`) defined in `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	$\oiiint\limits_0^1$	<code>\oiiint</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>\ointctrckwise</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	$\bot\limits_0^1$	<code>\bigbot</code>	LARGE UP TACK
U+027D9	$\top\limits_0^1$	<code>\bigtop</code>	LARGE DOWN TACK
U+029F8	$\big/$	<code>\xsol</code>	BIG SOLIDUS

U+029F9	\backslash	\xbsol	BIG REVERSE SOLIDUS
U+02A00	\odot	\bigodot	N-ARY CIRCLED DOT OPERATOR
U+02A01	\oplus	\bigoplus	N-ARY CIRCLED PLUS OPERATOR
U+02A02	\otimes	\bigotimes	N-ARY CIRCLED TIMES OPERATOR
U+02A03	$\cup\cdot$	\bigcupdot	N-ARY UNION OPERATOR WITH DOT
U+02A04	$\cup+$	\biguplus	N-ARY UNION OPERATOR WITH PLUS
U+02A05	\sqcap	\bigsqcap	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06	\sqcup	\bigsqcup	N-ARY SQUARE UNION OPERATOR
U+02A07	\wedge	\conjquant	TWO LOGICAL AND OPERATOR
U+02A08	\vee	\disjquant	TWO LOGICAL OR OPERATOR
U+02A09	\times	\bigtimes	N-ARY TIMES OPERATOR
U+02A0B	\int	\sumint	SUMMATION WITH INTEGRAL
U+02A0C	\iiint	\iiiint	QUADRUPLE INTEGRAL OPERATOR
U+02A0D	\int_0^1	\intbar	FINITE PART INTEGRAL
U+02A0E	\int_0^1	\intBar	INTEGRAL WITH DOUBLE STROKE
U+02A0F	\int_0^1	\fint	INTEGRAL AVERAGE WITH SLASH
U+02A10	\oint	\cirfnint	CIRCULATION FUNCTION
U+02A11	\int_0^1	\awint	ANTICLOCKWISE INTEGRATION LINE INTEGRATION WITH RECTANGULAR
U+02A12	\int_0^1	\rppolint	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13	\int_0^1	\scpolint	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE
U+02A14	\int_0^1	\npolint	POLE
U+02A15	\oint	\pointint	INTEGRAL AROUND A POINT OPERATOR
U+02A16	\int_0^1	\sqint	QUATERNION INTEGRAL OPERATOR INTEGRAL WITH LEFTWARDS ARROW WITH
U+02A17	\int_0^1	\intlarhk	HOOK
U+02A18	\int_0^1	\intx	INTEGRAL WITH TIMES SIGN
U+02A19	\int_0^1	\intcap	INTEGRAL WITH INTERSECTION
U+02A1A	\int_0^1	\intcup	INTEGRAL WITH UNION
U+02A1B	\int_0^1	\upint	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

`\l_um_nolimits_tl` 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:nNNn`). 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.

```

688 \tl_new:Nn \l_um_nolimits_tl {
689   \int\iint\iiint\iiint\oint\oiint\oiint
690   \intclockwise\varointclockwise\ointctr-clockwise\sumint
691   \intbar\intBar\oint\circfnint\awint\rppoint
692   \scpoint\npoint\pointint\sqint\intlarhk\intx
693   \intcap\intcup\upoint\lowint
694 }

```

`\addnolimits` This macro appends material to the macro containing the list of operators that don't take limits.

```

695 \DeclareDocumentCommand \addnolimits {m} {
696   \tl_put_right:Nn \l_um_nolimits_tl {#1}
697 }

```

`\removenolimits` Can this macro be given a better name? It removes an item from the nolimits list.

```

698 \DeclareDocumentCommand \removenolimits {m} {
699   \tl_remove_all_in:Nn \l_um_nolimits_tl {#1}
700 }

```

7.5 Radicals

The radical for square root is organised in `\um_set_mathsymbol:nNNn` 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.

```
701 \tl_new:Nn \l_um_radicals_tl {\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:

```
#####://####.#####-#####.###/###-###/#####2#####?##=#####/3853&#####/
3754
```















```
702 \let\left@primitive\left
```

```
703 \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+0221A	√	<code>\sqrt</code>	RADICAL
U+0221B	∛	<code>\cuberoot</code>	CUBE ROOT
U+0221C	∜	<code>\fourthroot</code>	FOURTH ROOT
U+02308	⌈	<code>\lceil</code>	LEFT CEILING
U+0230A	⌋	<code>\lfloor</code>	LEFT FLOOR
U+0231C	⌞	<code>\ulcorner</code>	UPPER LEFT CORNER
U+0231E	⏟	<code>\llcorner</code>	LOWER LEFT CORNER
			LIGHT LEFT TORTOISE SHELL BRACKET
U+02772	⎵	<code>\lbrbrak</code>	ORNAMENT
U+027C5	⎴	<code>\lbag</code>	LEFT S-SHAPED BAG DELIMITER
U+027CC	⎵	<code>\longdivision</code>	LONG DIVISION
			MATHEMATICAL LEFT WHITE SQUARE
U+027E6	⌈⌈	<code>\lBrack</code>	BRACKET
U+027E8	⌈	<code>\langle</code>	MATHEMATICAL LEFT ANGLE BRACKET
			MATHEMATICAL LEFT DOUBLE ANGLE
U+027EA	⌈⌈	<code>\lAngle</code>	BRACKET

			MATHEMATICAL LEFT WHITE TORTOISE
U+027EC		<code>\Lbrbrak</code>	SHELL BRACKET
U+02983		<code>\lBrace</code>	LEFT WHITE CURLY BRACKET
U+02985		<code>\lParen</code>	LEFT WHITE PARENTHESIS
U+02987		<code>\llparenthesis</code>	Z NOTATION LEFT IMAGE BRACKET
U+02989		<code>\llangle</code>	Z NOTATION LEFT BINDING BRACKET
U+0298B		<code>\lbrackubar</code>	LEFT SQUARE BRACKET WITH UNDERBAR
			LEFT SQUARE BRACKET WITH TICK IN TOP
U+0298D		<code>\lbrackultick</code>	CORNER LEFT SQUARE BRACKET WITH TICK IN
U+0298F		<code>\lbracklltick</code>	BOTTOM CORNER
U+02991		<code>\langedot</code>	LEFT ANGLE BRACKET WITH DOT
U+02993		<code>\lparenless</code>	LEFT ARC LESS-THAN BRACKET
U+02995		<code>\Lparenltr</code>	DOUBLE LEFT ARC GREATER-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+0007D	}	<code>\rbrace</code>	RIGHT CURLY BRACKET
U+02309	⌈	<code>\rceil</code>	RIGHT CEILING
U+0230B	⌋	<code>\rfloor</code>	RIGHT FLOOR
U+0231D	⌵	<code>\urcorner</code>	UPPER RIGHT CORNER
U+0231F	⌴	<code>\lrcorner</code>	LOWER RIGHT CORNER
			LIGHT RIGHT TORTOISE SHELL BRACKET
U+02773		<code>\rbrbrak</code>	ORNAMENT
U+027C6	⌋	<code>\rbag</code>	RIGHT S-SHAPED BAG DELIMITER
			MATHEMATICAL RIGHT WHITE SQUARE
U+027E7	⌋	<code>\rBrack</code>	BRACKET
U+027E9	⌋	<code>\rangle</code>	MATHEMATICAL RIGHT ANGLE BRACKET
			MATHEMATICAL RIGHT DOUBLE ANGLE
U+027EB	⌋	<code>\rAngle</code>	BRACKET
			MATHEMATICAL RIGHT WHITE TORTOISE
U+027ED		<code>\Rbrbrak</code>	SHELL BRACKET
U+02984	⌋	<code>\rBrace</code>	RIGHT WHITE CURLY BRACKET
U+02986	⌋	<code>\rParen</code>	RIGHT WHITE PARENTHESIS
U+02988	⌋	<code>\rrparenthesis</code>	Z NOTATION RIGHT IMAGE BRACKET
U+0298A	⌋	<code>\rrangle</code>	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+02996	⤵	\Rparenless	DOUBLE RIGHT ARC LESS-THAN BRACKET
U+02998)	\rblkbbrak	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	́	\acute	ACUTE ACCENT
U+00302	̂	\hat	CIRCUMFLEX ACCENT
U+00303	̃	\tilde	TILDE
U+00304	̄	\bar	MACRON
U+00305	̅	\overbar	OVERBAR EMBELLISHMENT
U+00306	̆	\breve	BREVE
U+00307	̇	\dot	DOT ABOVE
U+00308	̈	\ddot	DIERESIS
U+00309	̉	\ovhook	COMBINING HOOK ABOVE
U+0030A	̊	\ocirc	RING
U+0030C	̌	\check	CARON
U+00310	̎	\candra	CANDRABINDU (NON-SPACING)
U+00312	̐	\oturnedcomma	COMBINING TURNED COMMA ABOVE GREEK PSILI (SMOOTH BREATHING)
U+00313	̑	\osmooth	(NON-SPACING) GREEK DASIA (ROUGH BREATHING)
U+00314	̒	\orough	(NON-SPACING)
U+00315	̓	\ocommatopright	COMBINING COMMA ABOVE RIGHT
U+0031A	̔	\droang	LEFT ANGLE ABOVE (NON-SPACING) UNDER TILDE ACCENT (MULTIPLE)
U+00330	̘	\wideutilde	CHARACTERS AND NON-SPACING)
U+00331	̙	\underbar	COMBINING MACRON BELOW

U+00338		<code>\not</code>	COMBINING LONG SOLIDUS OVERLAY
U+020D0		<code>\leftharpoonaccent</code>	COMBINING LEFT HARPOON ABOVE
U+020D1		<code>\rightharpoonaccent</code>	COMBINING RIGHT HARPOON ABOVE
U+020D2		<code>\vertoverlay</code>	COMBINING LONG VERTICAL LINE OVERLAY
U+020D6		<code>\overleftarrow</code>	COMBINING LEFT ARROW ABOVE
U+020D7		<code>\vec</code>	COMBINING RIGHT ARROW ABOVE
U+020DB		<code>\dddot</code>	COMBINING THREE DOTS ABOVE
U+020DC		<code>\ddddot</code>	COMBINING FOUR DOTS ABOVE
U+020E1		<code>\overleftrightharpoon</code>	COMBINING LEFT RIGHT ARROW ABOVE
U+020E7		<code>\annuity</code>	COMBINING ANNUITY SYMBOL
U+020E8		<code>\threeunderdot</code>	COMBINING TRIPLE UNDERDOT
U+020E9		<code>\widebridgeabove</code>	COMBINING WIDE BRIDGE ABOVE COMBINING RIGHTWARDS HARPOON WITH
U+020EC		<code>\underrightharpoon</code>	BARB DOWNWARDS COMBINING LEFTWARDS HARPOON WITH
U+020ED		<code>\underleftharpoon</code>	BARB DOWNWARDS
U+020EE		<code>\underleftarrow</code>	COMBINING LEFT ARROW BELOW
U+020EF		<code>\underrightarrow</code>	COMBINING RIGHT ARROW BELOW
U+020F0		<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.

```

704 \newcommand\um@zf@feature[2]{
705   \define@key[zf]{options}{#1}[] {
706     \bool_if:NTF \l_um_fontspec_feature_bool {
707       #2
708     } {
709       \PackageError{fontspec/unicode-math}
710         {The ‘#1’ font feature can only be used for maths fonts}
711         {The feature you tried to use can only be in commands
712          like \protect\setmathfont}
713     }
714   }
715 }
```

8.1 OpenType maths font features

```

716 \um@zf@feature{ScriptStyle}{
717   \zf@update@ff{+ssty=0}
718 }
```

```

719 \um@zf@feature{ScriptScriptStyle}{
720   \zf@update@ff{+ssty=1}
721 }

```

8.2 Script and scriptscript font options

```

722 \define@cmdkey[um]{options}[um@]{script-features}{}
723 \define@cmdkey[um]{options}[um@]{sscript-features}{}
724 \define@cmdkey[um]{options}[um@]{script-font}{}
725 \define@cmdkey[um]{options}[um@]{sscript-font}{}

```

8.3 Range processing

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

```

726 \seq_new:N \g_um_mathalph_seq
727 \seq_new:N \l_um_mathalph_seq
728 \seq_new:N \l_um_char_range_seq
729 \define@choicekey+[um]{options}{range}[ \@tempa\@tempb]{ALL}{
730   \ifcase\@tempb\relax
731     \bool_set_true:N \l_um_init_bool
732   \fi
733 }{
734   \bool_set_false:N \l_um_init_bool
735   \seq_clear:N \l_um_char_range_seq
736   \seq_clear:N \l_um_mathalph_seq
737   \clist_map_inline:nn {#1} {
738     \um_if_mathalph_decl:nTF {##1} {
739       \seq_put_right:Nx \l_um_mathalph_seq { {\exp_not:V\l_um_tpa_tl} {\exp_not:V\l_um_tpb_tl} }
740     }{
741       \seq_put_right:Nn \l_um_char_range_seq {##1}
742     }
743   }
744 }
745 \prg_new_conditional:Nnn \um_if_mathalph_decl:n {TF} {
746   \tl_set:Nn \l_um_tpa_tl {#1}
747   \tl_set:Nn \l_um_tpb_tl {}
748   \tl_set:Nn \l_um_tpc_tl {}
749   \tl_if_in:NnT \l_um_tpa_tl {->} {
750     \exp_after:wN \um_split_arrow:w \l_um_tpa_tl \q_nil
751   }
752   \tl_if_in:NnT \l_um_tpa_tl {/} {
753     \exp_after:wN \um_split_slash:w \l_um_tpa_tl \q_nil
754   }
755   \seq_if_in:NVTF \g_um_mathalph_seq \l_um_tpa_tl {
756     \prg_return_true:
757   }{
758     \prg_return_false:
759   }

```

```

760 }
761 \cs_set:Npn \um_split_arrow:w #1->#2 \q_nil {
762   \tl_set:Nn \l_um_tmpa_tl {#1}
763   \tl_set:Nn \l_um_tmpe_tl {#2}
764 }
765 \cs_set:Npn \um_split_slash:w #1/#2 \q_nil {
766   \tl_set:Nn \l_um_tmpa_tl {#1}
767   \tl_set:Nn \l_um_tmpe_tl {#2}
768 }

```

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 `\l_um_char_range_seq`. 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:

```

769 \newcommand\um@parse@term[4]{
770   \seq_map_variable:NNn \l_um_char_range_seq \@ii {
771     \unless\ifx\@ii\empty
772       \@tempswafalse

```

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

```

773     \expandafter\um@firstchar\expandafter{\@ii}
774     \ifx\@tempa\um@backslash
775       \expandafter\ifx\@ii#2\relax
776       \@tempwatrue
777     \else
778       \expandafter\ifx\@ii#3\relax

```

```

779         \@tempwattrue
780         \fi
781     \fi

```

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

```

782     \else
783         \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
784     \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.

```

785     \if@tempswa
786         \ifx\um@char@num@range\@empty
787             \g@addto@macro\um@char@num@range{#1}
788         \else
789             \g@addto@macro\um@char@num@range{,#1}
790         \fi
791         #4%
792     \fi
793 \fi
794 }
795 }
796 \def\um@firstof#1#2\@nil{#1}
797 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
798 \def\um@firstchar#1{\edef\@tempa{\expandafter\um@firstof\string#1\@nil}}

```

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

```

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

```

Range	$r = x$
C-list input	<code>\@ii=X</code>
Macro input	<code>\um@parse@range X-\@marker-\@nil#1\@nil</code>
Arguments	$\textcolor{red}{\#1}-\textcolor{blue}{\#2}-\textcolor{blue}{\#3} = \textcolor{red}{X}-\textcolor{blue}{\@marker}-\{\}$

```

802     \expandafter\ifx\expandafter\@marker\@tempb\relax
803         \ifnum#4=#1\relax
804             \@tempwattrue
805         \fi
806     \else

```

Range	$r \geq x$
C-list input	<code>\@ii=X-</code>
Macro input	<code>\um@parse@range X--\@marker-\@nil#1\@nil</code>
Arguments	$\textcolor{red}{\#1}-\textcolor{blue}{\#2}-\textcolor{blue}{\#3} = \textcolor{red}{X}-\{\}-\textcolor{green}{\@marker}-$

```

807     \ifx\@empty\@tempb
808         \ifnum#4>\numexpr#1-1\relax

```

```

809         \@tempwattrue
810     \fi
811 \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-

```

812     \ifx\@empty\@tempa
813     \ifnum#4<\numexpr#2+1\relax
814     \@tempwattrue
815     \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-

```

816     \else
817     \ifnum#4>\numexpr#1-1\relax
818     \ifnum#4<\numexpr#2+1\relax
819     \@tempwattrue
820     \fi
821   \fi
822 \fi
823 \fi
824 \fi
825 }

```

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

```

826 \cs_set:Nn \um_map_chars_range:nnn {
827   \clist_map_inline:nn {#2} {
828     \prg_stepwise_inline:nnnn {0}{1}{#1} {
829       \um_map_char_internal:nn {##1+###1}{#3+###1}
830     }
831   }
832 }
833 \cs_new:Nn \um_map_char_noparse:nn {
834   \um_set_mathcode:nnnn
835   {\numexpr #1 \relax}{\mathalpha}{\um_symfont_tl}{\numexpr #2 \relax}
836 }
837 \cs_new:Nn \um_map_char_parse:nn {
838   \um@parse@term {#1} {\@nil} {\mathalpha} {
839     \um_map_char_noparse:nn {#1}{#2}
840   }

```

```

841 }
842 \cs_set:Nn \um_map_chars_xxvi:nn {
843   \um_map_chars_range:nnn {25}{#1}{#2}
844 }
845 \cs_set:Nn \um_map_chars_xxiii:nn {
846   \um_map_chars_range:nnn {24}{#1}{#2}
847 }
848 \cs_set:Nn \um_map_chars_x:nn {
849   \um_map_chars_range:nnn {9}{#1}{#2}
850 }
851 \cs_set:Nn \um_map_chars_Latin:nn {
852   \clist_map_inline:nn {#1} {
853     \um_map_chars_xxvi:cc { \um_to_usv:nn{##1}{Latin} }{ \um_to_usv:nn{#2}{Latin} }
854   }
855 }
856 \cs_set:Nn \um_map_chars_latin:nn {
857   \clist_map_inline:nn {#1} {
858     \um_map_chars_xxvi:cc {g_um_ ##1 _latin_usv}{g_um_ #2 _latin_usv}
859   }
860 }
861 \cs_set:Nn \um_map_chars_greek:nn {
862   \clist_map_inline:nn {#1} {
863     \um_map_chars_xxiii:cc {g_um_ ##1 _greek_usv}{g_um_ #2 _greek_usv}
864     \um_map_char:cc {g_um_ ##1 _varepsilon_usv}{g_um_ #2 _varepsilon_usv}
865     \um_map_char:cc {g_um_ ##1 _vartheta_usv }{g_um_ #2 _vartheta_usv }
866     \um_map_char:cc {g_um_ ##1 _varkappa_usv }{g_um_ #2 _varkappa_usv }
867     \um_map_char:cc {g_um_ ##1 _varphi_usv }{g_um_ #2 _varphi_usv }
868     \um_map_char:cc {g_um_ ##1 _varrho_usv }{g_um_ #2 _varrho_usv }
869     \um_map_char:cc {g_um_ ##1 _varpi_usv }{g_um_ #2 _varpi_usv }
870   }
871 }
872 \cs_set:Nn \um_map_chars_Greek:nn {
873   \clist_map_inline:nn {#1} {
874     \um_map_chars_xxiii:cc {g_um_ ##1 _Greek_usv}{g_um_ #2 _Greek_usv}
875     \um_map_char:cc {g_um_ ##1 _varTheta_usv}{g_um_ #2 _varTheta_usv}
876   }
877 }
878 \cs_set:Nn \um_map_chars_numbers:nn {
879   \um_map_chars_x:cc {g_um_#1_num_usv}{g_um_#2_num_usv}
880 }
881 \cs_set:Nn \um_map_char:nn {
882   \um_map_chars_range:nnn {0}{#1}{#2}
883 }
884 \cs_set:Nn \um_map_single:nnn {
885   \clist_map_inline:nn {#2} {
886     \um_map_char:cc {g_um_##1_#1_usv}{g_um_#3_#1_usv}

```

```

887 }
888 }
889 \cs_generate_variant:Nn \um_map_char:nn {cc}
890 \cs_generate_variant:Nn \um_map_chars_xxiii:nn {cc}
891 \cs_generate_variant:Nn \um_map_chars_xxvi:nn {cc}
892 \cs_generate_variant:Nn \um_map_chars_x:nn {cc}

\um_set_mathalphabet_char:Nnn #1 : Maths alphabet
#2 : Input char(s)
#3 : Output char
Loops through character ranges setting \mathcode.
893 \cs_set:Npn \exp_args:Nnff {\:n\::f\::f\:::}
894 \cs_new:Nn \um_set_mathalphabet_char:Nnn {
895   \clist_map_variable:nNn {#2} \l_um_input_num {
896     \exp_args:Nnff \um_mathmap:Nnn {#1}
897       {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
898   }
899 }

```

```

\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.
900 \cs_new:Nn \um_set_mathalph_range:nNnn {
901   \clist_map_variable:nNn {#3} \l_um_input_num {
902     \errorcontextlines=999
903     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
904       \exp_args:Nnff \um_mathmap:Nnn {#2}
905         {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
906         {\number\numexpr \l_um_inc_num + #4 \relax}
907     }
908   }
909 }
910 \cs_new:Nn \um_set_mathalphabet_x:Nnn {
911   \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
912 }
913 \cs_new:Nn \um_set_mathalphabet_xxvi:Nnn {
914   \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
915 }
916 \cs_new:Nn \um_set_mathalphabet_xxiii:Nnn {
917   \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
918 }
919
920 \cs_new:Nn \um_set_mathalphabet_pos:Nnnn {
921   \cs_if_exist:cT {g_um_#4_#2_usv} {
922     \clist_map_inline:nn {#3} {

```



```

923     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_#2_usv}{g_um_#4_#2_usv}
924   }
925 }
926 }
927 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {
928   \clist_map_inline:nn {#2} {
929     \um_set_mathalphabet_x:Ncc #1 {g_um_##1_num_usv}{g_um_#3_num_usv}
930   }
931 }
932 \cs_new:Nn \um_set_mathalphabet_Latin:Nnn {
933   \clist_map_inline:nn {#2} {
934     \um_set_mathalphabet_xxvi:Ncc #1 {g_um_##1_Latin_usv}{g_um_#3_Latin_usv}
935   }
936 }
937 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
938   \clist_map_inline:nn {#2} {
939     \um_set_mathalphabet_xxvi:Ncc #1 {g_um_##1_latin_usv}{g_um_#3_latin_usv}
940     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_h_usv} {g_um_#3_h_usv}
941   }
942 }
943 \cs_new:Nn \um_set_mathalphabet_Greek:Nnn {
944   \clist_map_inline:nn {#2} {
945     \um_set_mathalphabet_xxiii:Ncc #1 {g_um_##1_Greek_usv} {g_um_#3_Greek_usv}
946     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_varTheta_usv}{g_um_#3_varTheta_usv}
947   }
948 }
949 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
950   \clist_map_inline:nn {#2} {
951     \um_set_mathalphabet_xxiii:Ncc #1 {g_um_##1_greek_usv} {g_um_#3_greek_usv}
952     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_varepsilon_usv}{g_um_#3_varepsilon_usv}
953     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_vartheta_usv} {g_um_#3_vartheta_usv}
954     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_varkappa_usv} {g_um_#3_varkappa_usv}
955     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_varphi_usv} {g_um_#3_varphi_usv}
956     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_varrho_usv} {g_um_#3_varrho_usv}
957     \um_set_mathalphabet_char:Ncc #1 {g_um_##1_varpi_usv} {g_um_#3_varpi_usv}
958   }
959 }
960 \cs_generate_variant:Nn \um_set_mathalphabet_char:Nnn {Ncc}
961 \cs_generate_variant:Nn \um_set_mathalphabet_xxiii:Nnn {Ncc}
962 \cs_generate_variant:Nn \um_set_mathalphabet_xxvi:Nnn {Ncc}
963 \cs_generate_variant:Nn \um_set_mathalphabet_x:Nnn {Ncc}

```

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.

```

964 \AtBeginDocument{\um_resolve_greek:}
965 \cs_new:Nn \um_resolve_greek: {
966   \clist_map_inline:nn {
967     Alpha,Beta,Gamma,Delta,Epsilon,Zeta,Eta,Theta,Iota,Kappa,Lambda,
968     alpha,beta,gamma,delta,          zeta,eta,theta,iota,kappa,lambda,
969     Mu,Nu,Xi,Omicron,Pi,Rho,Sigma,Tau,Upsilon,Phi,Chi,Psi,Omega,
970     mu,nu,xi,omicron,pi,rho,sigma,tau,upsilon,    chi,psi,omega,
971     varTheta,
972     varsigma,vartheta,varkappa,varrho,varpi
973   }{
974     \tl_set:cx {##1} { \exp_not:c { mit ##1 } }
975   }
976   \tl_set:Nn \epsilon {
977     \bool_if:NTF \g_um_texgreek_bool \mitvarepsilon \mitepsilon
978   }
979   \tl_set:Nn \phi {
980     \bool_if:NTF \g_um_texgreek_bool \mitvarphi \mitphi
981   }
982   \tl_set:Nn \varepsilon {
983     \bool_if:NTF \g_um_texgreek_bool \mitepsilon \mitvarepsilon
984   }
985   \tl_set:Nn \varphi {
986     \bool_if:NTF \g_um_texgreek_bool \mitphi \mitvarphi
987   }
988 }

```

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 each alphabet to detect if the font supports each alphabet shape.
- 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.

9.0.1 Macros

This is every math alphabet known to unicode-math:

`\g_um_mathalph_seq`

```

989 \seq_clear:N \g_um_mathalph_seq
990 \tl_map_inline:nn {
991   \mathup\mathit
992   \mathbb\mathscr\mathfrak\mathtt
993   \mathsf\mathsfup\mathsfit
994   \mathbf\mathbfup\mathbfit
995   \mathbfscr\mathbffrak
996   \mathbfsf\mathbfsfup\mathbfsfit
997 }{
998   \seq_put_right:Nn \g_um_mathalph_seq {#1}
999 }
```

`\um_setup_alphabets:`

```

1000
1001 \tl_new:Nn \g_um_mathup_alph_clist {latin, Latin, greek, Greek, num}
1002 \tl_new:Nn \g_um_mathit_alph_clist {latin, Latin, greek, Greek}
1003 \tl_new:Nn \g_um_mathscr_alph_clist {latin, Latin}
1004 \tl_new:Nn \g_um_mathfrak_alph_clist {latin, Latin}
1005 \tl_new:Nn \g_um_mathbfscr_alph_clist {latin, Latin}
1006 \tl_new:Nn \g_um_mathbffrak_alph_clist {latin, Latin}
1007 \tl_new:Nn \g_um_mathbb_alph_clist {latin, Latin, num}
1008 \tl_new:Nn \g_um_mathtt_alph_clist {latin, Latin, num}
1009 \tl_new:Nn \g_um_mathsf_alph_clist {latin, Latin, num}
1010 \tl_new:Nn \g_um_mathsfup_alph_clist {latin, Latin, num}
1011 \tl_new:Nn \g_um_mathsfit_alph_clist {latin, Latin}
1012 \tl_new:Nn \g_um_mathbf_alph_clist {latin, Latin, greek, Greek, num}
1013 \tl_new:Nn \g_um_mathbfup_alph_clist {latin, Latin, greek, Greek, num}
1014 \tl_new:Nn \g_um_mathbfit_alph_clist {latin, Latin, greek, Greek, num}
1015 \tl_new:Nn \g_um_mathbfsf_alph_clist {latin, Latin, greek, Greek, num}
1016 \tl_new:Nn \g_um_mathbfsfup_alph_clist {latin, Latin, greek, Greek, num}
1017 \tl_new:Nn \g_um_mathbfsfit_alph_clist {latin, Latin, greek, Greek}
1018
1019 \tl_new:Nn \g_um_mathup_latin_usv {\a-\z}
```

```

1020 \tl_new:Nn \g_um_mathup_Latin_usv {\A-`Z}
1021 \tl_new:Nn \g_um_mathup_greek_usv {"3B1-"3C9,"3F5,"3D1,"3F0,"3D5,"3F1,"3D6,"3DD}
1022 \tl_new:Nn \g_um_mathup_Greek_usv {"391-"3A9,"3F4,"3DC}
1023 \tl_new:Nn \g_um_mathup_num_usv {\0-`9}
1024
1025 \tl_new:Nn \g_um_mathit_latin_usv {"1D44E-"1D467,\g_um_it_h_usv}
1026 \tl_new:Nn \g_um_mathit_Latin_usv {"1D434-"1D44C}
1027 \tl_new:Nn \g_um_mathit_greek_usv {"1D6FC-"1D714,"1D716-1D71B}
1028 \tl_new:Nn \g_um_mathit_Greek_usv {"1D6E2-"1D6FA}
1029
1030 \seq_new:N \l_um_missing_alph_seq
1031 \cs_new:Nn \um_setup_alphabets: {
1032   \seq_clear:N \l_um_missing_alph_seq
1033   \seq_if_empty:NTF \l_um_mathalph_seq {
1034     \um_setup_math_alphabet:NV \mathup \g_um_mathup_alph_clist
1035     \um_setup_math_alphabet:NV \mathit \g_um_mathit_alph_clist
1036     \um_setup_math_alphabet:NV \mathbb \g_um_mathbb_alph_clist
1037     \um_setup_math_alphabet:NV \mathscr \g_um_mathscr_alph_clist
1038     \um_setup_math_alphabet:NV \mathfrak \g_um_mathfrak_alph_clist
1039     \um_setup_math_alphabet:NV \mathsf \g_um_mathsf_alph_clist
1040     \um_setup_math_alphabet:NV \mathsfup \g_um_mathsfup_alph_clist
1041     \um_setup_math_alphabet:NV \mathsfit \g_um_mathsfit_alph_clist
1042     \um_setup_math_alphabet:NV \mathtt \g_um_mathtt_alph_clist
1043     \um_setup_math_alphabet:NV \mathbf \g_um_mathbf_alph_clist
1044     \um_setup_math_alphabet:NV \mathbfup \g_um_mathbfup_alph_clist
1045     \um_setup_math_alphabet:NV \mathbfit \g_um_mathbfit_alph_clist
1046     \um_setup_math_alphabet:NV \mathbfscr \g_um_mathbfscr_alph_clist
1047     \um_setup_math_alphabet:NV \mathbffrak \g_um_mathbffrak_alph_clist
1048     \um_setup_math_alphabet:NV \mathbfsf \g_um_mathbfsf_alph_clist
1049     \um_setup_math_alphabet:NV \mathbfsfup \g_um_mathbfsfup_alph_clist
1050     \um_setup_math_alphabet:NV \mathbfsfit \g_um_mathbfsfit_alph_clist
1051     \um_setup_math_mapping:n {up }
1052     \um_setup_math_mapping:n {it }
1053     \um_setup_math_mapping:n {bb }
1054     \um_maybe_init_alphabet:n {bbit }
1055     \um_setup_math_mapping:n {bbit }
1056     \um_setup_math_mapping:n {bfup }
1057     \um_setup_math_mapping:n {bfit }
1058     \um_setup_math_mapping:n {bfsfup}
1059     \um_setup_math_mapping:n {bfsfit}
1060     \seq_if_empty:NF \l_um_missing_alph_seq {
1061       \typeout{
1062         Package~unicode-math~Warning:~
1063         missing~math~alphabets~in~font~ \fontname\l_um_font
1064       }
1065       \seq_map_inline:Nn \l_um_missing_alph_seq {

```

```

1066         \typeout{\space\space\space\space##1}
1067     }
1068 }
1069 }{
1070 \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
1071 \seq_map_inline:Nn \l_um_mathalph_seq {
1072     \tl_set:No \l_um_tmpa_tl { \use_i:nnn ##1 }
1073     \tl_set:No \l_um_tmpb_tl { \use_ii:nnn ##1 }
1074     \tl_set:No \l_um_tmpc_tl { \use_iii:nnn ##1 }
1075     \tl_if_empty:NF \l_um_tmpc_tl {
1076         \PackageWarning{unicode-math}{alphabet~remapping~not~yet~implemented}
1077     }
1078     \tl_if_empty:NT \l_um_tmpb_tl {
1079         \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
1080         \tl_set:Nv \l_um_tmpb_tl { g_um_ \exp_after:wN \cs_to_str:N \l_um_tmpa_tl _alph_clist }
1081     }
1082     \um_setup_math_alphabet:VV \l_um_tmpa_tl \l_um_tmpb_tl
1083 }
1084 }
1085 }

```

`\um_setup_math_alphabet:Nn` **#1** : Math font family name (e.g., `\mathbb`)
#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 loop through them defining the individual ranges.

```

1086 \cs_new:Nn \um_setup_math_alphabet:Nn {
1087     \tl_set:Nx \l_um_tmpa_tl {\cs_to_str:N #1}
1088     \tl_set:Nx \l_um_tmpb_tl {\exp_after:wN \use_none:nnnn \l_um_tmpa_tl}
1089     \clist_map_inline:nn {#2} {
1090         \um_glyph_if_exist:cT {g_um_ \l_um_tmpb_tl _##1_usv}{
1091             \exp_args:NV \um_maybe_init_alphabet:n \l_um_tmpb_tl
1092             \clist_map_break:
1093         }
1094     }
1095     \clist_map_inline:nn {#2} {
1096         \um_glyph_if_exist:cTF {g_um_ \l_um_tmpb_tl _##1_usv}{
1097             \use:c {um_config_ \l_um_tmpa_tl _##1:}
1098         }{
1099             \seq_put_right:Nx \l_um_missing_alph_seq {
1100                 \@backslashchar
1101                 \l_um_tmpa_tl\space(\tl_use:c{g_um_math_alphabet_name_##1_tl})
1102             }
1103         }
1104     }
1105 }
1106 \cs_generate_variant:Nn \um_setup_math_alphabet:Nn {NV,VV}

```

```

1107
1108 \tl_set:Nn \g_um_math_alphabet_name_latin_tl {Latin,~lowercase}
1109 \tl_set:Nn \g_um_math_alphabet_name_Latin_tl {Latin,~uppercase}
1110 \tl_set:Nn \g_um_math_alphabet_name_greek_tl {Greek,~lowercase}
1111 \tl_set:Nn \g_um_math_alphabet_name_Greek_tl {Greek,~uppercase}
1112 \tl_set:Nn \g_um_math_alphabet_name_num_tl {Numerals}
1113 \cs_new:Nn \um_setup_math_mapping:n {
1114   \cs_if_exist:cT {um_setup_math#1:} {
1115     \use:c {um_config_math#1_misc:}
1116   }
1117 }

1118 \cs_set:Nn \um_init_alphabet:n {
1119   \wlog{unicode-math:~Initialiasing~\@backslashchar math#1}
1120   \um_prepare_alph:n {#1}
1121   \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
1122 }

```

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

```

1123 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
1124   \etex_iffontchar:D \l_um_font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi
1125 }
1126 \cs_generate_variant:Nn \um_glyph_if_exist_p:n {c}
1127 \cs_generate_variant:Nn \um_glyph_if_exist:nTF {c}
1128 \cs_generate_variant:Nn \um_glyph_if_exist:nT {c}
1129 \cs_generate_variant:Nn \um_glyph_if_exist:nF {c}

```

`\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.

```

1130 \cs_new:Nn \um_prepare_alph:n {
1131   \cs_if_exist:cF {um_math#1:n} {
1132     \cs_set:cpn {um_math#1:n} ##1 {
1133       \use:c {um_setup_math#1:} ##1 \egroup
1134     }
1135     \cs_set_protected:cpn {math#1} {
1136       \bgroup
1137       \mode_if_math:F {
1138         \egroup\expandafter
1139         \non@alpherr\expandafter{\csname math#1\endcsname\space}
1140       }
1141       \use:c {um_math#1:n}
1142     }
1143   }
1144 }

```

9.1 Alphabets

9.1.1 Upright: \mathup

```
1145 \cs_new:Npn \um_config_mathup_num: {  
1146   \um_map_chars_numbers:nn {up}{up}  
1147   \um_set_mathalphabet_numbers:Nnn \mathup {up}{up}  
1148 }  
1149 \cs_new:Npn \um_config_mathup_Latin: {  
1150   \bool_if:NTF \g_um_literal_bool {  
1151     \um_map_chars_Latin:nn {up} {up}  
1152   }{  
1153     \bool_if:NT \g_um_upLatin_bool {  
1154       \um_map_chars_Latin:nn {up,it} {up}  
1155     }  
1156   }  
1157   \um_set_mathalphabet_Latin:Nnn \mathup {up,it}{up}  
1158 }  
1159 \cs_new:Npn \um_config_mathup_latin: {  
1160   \bool_if:NTF \g_um_literal_bool {  
1161     \um_map_chars_latin:nn {up} {up}  
1162   }{  
1163     \bool_if:NT \g_um_uplatin_bool {  
1164       \um_map_chars_latin:nn {up,it} {up}  
1165       \um_map_single:nnn {h}{up,it}{up}  
1166     }  
1167   }  
1168   \um_set_mathalphabet_latin:Nnn \mathup {up,it}{up}  
1169 }  
1170 \cs_new:Npn \um_config_mathup_Greek: {  
1171   \bool_if:NTF \g_um_literal_bool {  
1172     \um_map_chars_Greek:nn {up}{up}  
1173   }{  
1174     \bool_if:NT \g_um_upGreek_bool {  
1175       \um_map_chars_Greek:nn {up,it}{up}  
1176     }  
1177   }  
1178   \um_set_mathalphabet_Greek:Nnn \mathup {up,it}{up}  
1179 }  
1180 \cs_new:Npn \um_config_mathup_greek: {  
1181   \bool_if:NTF \g_um_literal_bool {  
1182     \um_map_chars_greek:nn {up} {up}  
1183   }{  
1184     \bool_if:NT \g_um_upgreek_bool {  
1185       \um_map_chars_greek:nn {up,it} {up}  
1186     }  
1187   }  
}
```

```

1188 \um_set_mathalphabet_greek:Nnn \mathup {up,it} {up}
1189 }
1190 \cs_new:Npn \um_config_mathup_misc: {
1191   \um_set_mathalphabet_pos:Nnnn \mathup {partial} {up,it}{up}
1192   \um_set_mathalphabet_pos:Nnnn \mathup {Nabla} {up,it}{up}
1193 }

```

9.1.2 Italic: \mathit

```

1194 \cs_new:Npn \um_config_mathit_Latin: {
1195   \bool_if:NTF \g_um_literal_bool {
1196     \um_map_chars_Latin:nn {it} {it}
1197   }{
1198     \bool_if:NF \g_um_upLatin_bool {
1199       \um_map_chars_Latin:nn {up,it} {it}
1200     }
1201   }
1202   \um_set_mathalphabet_Latin:Nnn \mathit {up,it}{it}
1203 }
1204 \cs_new:Npn \um_config_mathit_latin: {
1205   \bool_if:NTF \g_um_literal_bool {
1206     \um_map_chars_latin:nn {it} {it}
1207     \um_map_single:nnn {h}{it}{it}
1208   }{
1209     \bool_if:NF \g_um_uplatin_bool {
1210       \um_map_chars_latin:nn {up,it} {it}
1211       \um_map_single:nnn {h}{up,it}{it}
1212     }
1213   }
1214   \um_set_mathalphabet_latin:Nnn \mathit {up,it}{it}
1215 }
1216 \cs_new:Npn \um_config_mathit_Greek: {
1217   \bool_if:NTF \g_um_literal_bool {
1218     \um_map_chars_Greek:nn {it}{it}
1219   }{
1220     \bool_if:NF \g_um_upGreek_bool {
1221       \um_map_chars_Greek:nn {up,it}{it}
1222     }
1223   }
1224   \um_set_mathalphabet_Greek:Nnn \mathit {up,it}{it}
1225 }
1226 \cs_new:Npn \um_config_mathit_greek: {
1227   \bool_if:NTF \g_um_literal_bool {
1228     \um_map_chars_greek:nn {it} {it}
1229   }{
1230     \bool_if:NF \g_um_upgreek_bool {
1231       \um_map_chars_greek:nn {it,up} {it}

```



```

1232     }
1233   }
1234   \um_set_mathalphabet_greek:Nnn \mathit {up,it} {it}
1235 }
1236 \cs_new:Npn \um_config_mathit_misc: {
1237   \um_set_mathalphabet_pos:Nnnn \mathit {partial} {up,it}{it}
1238   \um_set_mathalphabet_pos:Nnnn \mathit {Nabla} {up,it}{it}
1239 }

```

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

```

1240 \cs_new:Npn \um_config_mathbb_latin: {
1241   \um_set_mathalphabet_latin:Nnn \mathbb {up,it}{bb}
1242 }
1243 \cs_new:Npn \um_config_mathbb_Latin: {
1244   \um_set_mathalphabet_Latin:Nnn \mathbb {up,it}{bb}
1245   \um_set_mathalphabet_pos:Nnnn \mathbb {C} {up,it} {bb}
1246   \um_set_mathalphabet_pos:Nnnn \mathbb {H} {up,it} {bb}
1247   \um_set_mathalphabet_pos:Nnnn \mathbb {N} {up,it} {bb}
1248   \um_set_mathalphabet_pos:Nnnn \mathbb {P} {up,it} {bb}
1249   \um_set_mathalphabet_pos:Nnnn \mathbb {Q} {up,it} {bb}
1250   \um_set_mathalphabet_pos:Nnnn \mathbb {R} {up,it} {bb}
1251   \um_set_mathalphabet_pos:Nnnn \mathbb {Z} {up,it} {bb}
1252 }
1253 \cs_new:Npn \um_config_mathbb_num: {
1254   \um_set_mathalphabet_numbers:Nnn \mathbb {up}{bb}
1255 }
1256 \cs_new:Npn \um_config_mathbb_misc: {
1257   \um_set_mathalphabet_pos:Nnnn \mathbb {Pi} {up,it} {bb}
1258   \um_set_mathalphabet_pos:Nnnn \mathbb {pi} {up,it} {bb}
1259   \um_set_mathalphabet_pos:Nnnn \mathbb {Gamma} {up,it} {bb}
1260   \um_set_mathalphabet_pos:Nnnn \mathbb {gamma} {up,it} {bb}
1261   \um_set_mathalphabet_pos:Nnnn \mathbb {summation} {up} {bb}
1262 }
1263 \cs_new:Npn \um_config_mathbbit_misc: {
1264   \um_set_mathalphabet_pos:Nnnn \mathbbit {D} {up,it} {bbit}
1265   \um_set_mathalphabet_pos:Nnnn \mathbbit {d} {up,it} {bbit}
1266   \um_set_mathalphabet_pos:Nnnn \mathbbit {e} {up,it} {bbit}
1267   \um_set_mathalphabet_pos:Nnnn \mathbbit {i} {up,it} {bbit}
1268   \um_set_mathalphabet_pos:Nnnn \mathbbit {j} {up,it} {bbit}
1269 }

```

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

```

1270 \cs_new:Npn \um_config_mathscr_Latin: {
1271   \um_set_mathalphabet_Latin:Nnn \mathscr {up,it}{scr}
1272   \um_set_mathalphabet_pos:Nnnn \mathscr {B}{up,it}{scr}
1273   \um_set_mathalphabet_pos:Nnnn \mathscr {E}{up,it}{scr}
1274   \um_set_mathalphabet_pos:Nnnn \mathscr {F}{up,it}{scr}

```

```

1275 \um_set_mathalphabet_pos:Nnnn \mathscr {H}{up,it}{scr}
1276 \um_set_mathalphabet_pos:Nnnn \mathscr {I}{up,it}{scr}
1277 \um_set_mathalphabet_pos:Nnnn \mathscr {L}{up,it}{scr}
1278 \um_set_mathalphabet_pos:Nnnn \mathscr {M}{up,it}{scr}
1279 \um_set_mathalphabet_pos:Nnnn \mathscr {R}{up,it}{scr}
1280 }
1281 \cs_new:Npn \um_config_mathscr_latin: {
1282 \um_set_mathalphabet_latin:Nnn \mathscr {up,it}{scr}
1283 \um_set_mathalphabet_pos:Nnnn \mathscr {e}{up,it}{scr}
1284 \um_set_mathalphabet_pos:Nnnn \mathscr {g}{up,it}{scr}
1285 \um_set_mathalphabet_pos:Nnnn \mathscr {o}{up,it}{scr}
1286 }

```

9.1.5 Fraktur or fraktur or blackletter: `\mathfrak`

```

1287 \cs_new:Npn \um_config_mathfrak_Latin: {
1288 \um_set_mathalphabet_Latin:Nnn \mathfrak {up,it}{frak}
1289 \um_set_mathalphabet_pos:Nnnn \mathfrak {C}{up,it}{frak}
1290 \um_set_mathalphabet_pos:Nnnn \mathfrak {H}{up,it}{frak}
1291 \um_set_mathalphabet_pos:Nnnn \mathfrak {I}{up,it}{frak}
1292 \um_set_mathalphabet_pos:Nnnn \mathfrak {R}{up,it}{frak}
1293 \um_set_mathalphabet_pos:Nnnn \mathfrak {Z}{up,it}{frak}
1294 }
1295 \cs_new:Npn \um_config_mathfrak_latin: {
1296 \um_set_mathalphabet_latin:Nnn \mathfrak {up,it}{frak}
1297 }

```

9.1.6 Sans serif upright: `\mathsfup`

```

1298 \cs_new:Npn \um_config_mathsfup_num: {
1299 \um_set_mathalphabet_numbers:Nnn \mathsf {up}{sf}
1300 \um_set_mathalphabet_numbers:Nnn \mathsfup {up}{sf}
1301 }
1302 \cs_new:Npn \um_config_mathsfup_Latin: {
1303 \bool_if:NTF \g_um_sfliteral_bool {
1304 \um_map_chars_Latin:nn {sfup} {sfup}
1305 \um_set_mathalphabet_Latin:Nnn \mathsf {up}{sfup}
1306 }{
1307 \bool_if:NT \g_um_upsans_bool {
1308 \um_map_chars_Latin:nn {sfup,sfit} {sfup}
1309 \um_set_mathalphabet_Latin:Nnn \mathsf {up,it}{sfup}
1310 }
1311 }
1312 \um_set_mathalphabet_Latin:Nnn \mathsfup {up,it}{sfup}
1313 }
1314 \cs_new:Npn \um_config_mathsfup_latin: {
1315 \bool_if:NTF \g_um_sfliteral_bool {
1316 \um_map_chars_latin:nn {sfup} {sfup}
1317 \um_set_mathalphabet_latin:Nnn \mathsf {up}{sfup}

```

```

1318 }{
1319   \bool_if:NT \g_um_upsans_bool {
1320     \um_map_chars_latin:nn {sfup,sfit} {sfup}
1321     \um_set_mathalphabet_latin:Nnn \mathsf {up,it}{sfup}
1322   }
1323 }
1324 \um_set_mathalphabet_latin:Nnn \mathsfup {up,it}{sfup}
1325 }

```

9.1.7 Sans serif italic: \mathsfit

```

1326 \cs_new:Npn \um_config_mathsf_Latin: {
1327   \bool_if:NTF \g_um_sfliteral_bool {
1328     \um_map_chars_Latin:nn {sfit} {sfit}
1329     \um_set_mathalphabet_Latin:Nnn \mathsf {it}{sfit}
1330   }{
1331     \bool_if:NF \g_um_upsans_bool {
1332       \um_map_chars_Latin:nn {sfup,sfit} {sfit}
1333       \um_set_mathalphabet_Latin:Nnn \mathsf {up,it}{sfit}
1334     }
1335   }
1336   \um_set_mathalphabet_Latin:Nnn \mathsfit {up,it}{sfit}
1337 }
1338 \cs_new:Npn \um_config_mathsf_latin: {
1339   \bool_if:NTF \g_um_sfliteral_bool {
1340     \um_map_chars_latin:nn {sfit} {sfit}
1341     \um_set_mathalphabet_latin:Nnn \mathsf {it}{sfit}
1342   }{
1343     \bool_if:NF \g_um_upsans_bool {
1344       \um_map_chars_latin:nn {sfup,sfit} {sfit}
1345       \um_set_mathalphabet_latin:Nnn \mathsf {up,it}{sfit}
1346     }
1347   }
1348   \um_set_mathalphabet_latin:Nnn \mathsfit {up,it}{sfit}
1349 }

```

9.1.8 Typewriter or monospaced: \mathtt

```

1350 \cs_new:Npn \um_config_mathtt_num: {
1351   \um_set_mathalphabet_numbers:Nnn \mathtt {up}{tt}
1352 }
1353 \cs_new:Npn \um_config_mathtt_Latin: {
1354   \um_set_mathalphabet_Latin:Nnn \mathtt {up,it}{tt}
1355 }
1356 \cs_new:Npn \um_config_mathtt_latin: {
1357   \um_set_mathalphabet_latin:Nnn \mathtt {up,it}{tt}
1358 }

```

9.1.9 Bold Italic: \mathbfit

```

1359 \cs_new:Npn \um_config_mathbfit_Latin: {
1360   \bool_if:NF \g_um_bfupLatin_bool {
1361     \um_map_chars_Latin:nn {bfup,bfit} {bfup}
1362   }
1363   \um_set_mathalphabet_Latin:Nnn \mathbfit {up,it}{bfit}
1364   \bool_if:NTF \g_um_bfliteral_bool {
1365     \um_map_chars_Latin:nn {bfit} {bfit}
1366     \um_set_mathalphabet_Latin:Nnn \mathbf {it}{bfit}
1367   }{
1368     \bool_if:NF \g_um_bfupLatin_bool {
1369       \um_map_chars_Latin:nn {bfup,bfit} {bfit}
1370       \um_set_mathalphabet_Latin:Nnn \mathbf {up,it}{bfit}
1371     }
1372   }
1373 }
1374 \cs_new:Npn \um_config_mathbfit_latin: {
1375   \bool_if:NF \g_um_bfuplatin_bool {
1376     \um_map_chars_latin:nn {bfup,bfit} {bfit}
1377   }
1378   \um_set_mathalphabet_latin:Nnn \mathbfit {up,it}{bfit}
1379   \bool_if:NTF \g_um_bfliteral_bool {
1380     \um_map_chars_latin:nn {bfit} {bfit}
1381     \um_set_mathalphabet_latin:Nnn \mathbf {it}{bfit}
1382   }{
1383     \bool_if:NF \g_um_bfuplatin_bool {
1384       \um_map_chars_latin:nn {bfup,bfit} {bfit}
1385       \um_set_mathalphabet_latin:Nnn \mathbf {up,it}{bfit}
1386     }
1387   }
1388 }
1389 \cs_new:Npn \um_config_mathbfit_Greek: {
1390   \um_set_mathalphabet_Greek:Nnn \mathbfit {up,it}{bfit}
1391   \bool_if:NTF \g_um_bfliteral_bool {
1392     \um_map_chars_Greek:nn {bfit}{bfit}
1393     \um_set_mathalphabet_Greek:Nnn \mathbf {it}{bfit}
1394   }{
1395     \bool_if:NF \g_um_bfupGreek_bool {
1396       \um_map_chars_Greek:nn {bfup,bfit}{bfit}
1397       \um_set_mathalphabet_Greek:Nnn \mathbf {up,it}{bfit}
1398     }
1399   }
1400 }
1401 \cs_new:Npn \um_config_mathbfit_greek: {
1402   \um_set_mathalphabet_greek:Nnn \mathbfit {up,it} {bfit}
1403   \bool_if:NTF \g_um_bfliteral_bool {
1404     \um_map_chars_greek:nn {bfit} {bfit}

```

```

1405 \um_set_mathalphabet_greek:Nnn \mathbfit {it} {bfit}
1406 }{
1407 \bool_if:NF \g_um_bfupgreek_bool {
1408 \um_map_chars_greek:nn {bfit,bfup} {bfit}
1409 }
1410 \bool_if:NF \g_um_bfupgreek_bool {
1411 \um_set_mathalphabet_greek:Nnn \mathbfit {up,it} {bfit}
1412 }
1413 }
1414 }
1415 \cs_new:Npn \um_config_mathbfit_misc: {
1416 \um_set_mathalphabet_pos:Nnnn \mathbfit {partial} {up,it}{bfit}
1417 \um_set_mathalphabet_pos:Nnnn \mathbfit {Nabla} {up,it}{bfit}
1418 \bool_if:NTF \g_um_bfliteral_bool {
1419 \um_set_mathalphabet_pos:Nnnn \mathbf {partial} {it}{bfit}
1420 \um_set_mathalphabet_pos:Nnnn \mathbf {Nabla} {it}{bfit}
1421 }{
1422 \bool_if:NF \g_um_upNabla_bool {
1423 \um_set_mathalphabet_pos:Nnnn \mathbf {Nabla} {up,it}{bfit}
1424 }
1425 \bool_if:NF \g_um_uppartial_bool {
1426 \um_set_mathalphabet_pos:Nnnn \mathbf {partial} {up,it}{bfit}
1427 }
1428 }
1429 }

```

9.1.10 Bold Upright: `\mathbfup`

```

1430 \cs_new:Npn \um_config_mathbfup_num: {
1431 \um_set_mathalphabet_numbers:Nnn \mathbf {up}{bfup}
1432 \um_set_mathalphabet_numbers:Nnn \mathbfup {up}{bfup}
1433 }
1434 \cs_new:Npn \um_config_mathbfup_Latin: {
1435 \bool_if:NT \g_um_bfupLatin_bool {
1436 \um_map_chars_Latin:nn {bfup,bfit} {bfit}
1437 }
1438 \um_set_mathalphabet_Latin:Nnn \mathbfup {up,it}{bfup}
1439 \bool_if:NTF \g_um_bfliteral_bool {
1440 \um_map_chars_Latin:nn {bfup} {bfup}
1441 \um_set_mathalphabet_Latin:Nnn \mathbf {up}{bfup}
1442 }{
1443 \bool_if:NT \g_um_bfupLatin_bool {
1444 \um_map_chars_Latin:nn {bfup,bfit} {bfup}
1445 \um_set_mathalphabet_Latin:Nnn \mathbf {up,it}{bfup}
1446 }
1447 }
1448 }

```

```

1449 \cs_new:Npn \um_config_mathbfup_latin: {
1450   \bool_if:NT \g_um_bfuplatin_bool {
1451     \um_map_chars_latin:nn {bfup,bfit} {bfup}
1452   }
1453   \um_set_mathalphabet_latin:Nnn \mathbfup {up,it}{bfup}
1454   \bool_if:NTF \g_um_bfliteral_bool {
1455     \um_map_chars_latin:nn {bfup} {bfup}
1456     \um_set_mathalphabet_latin:Nnn \mathbf {up}{bfup}
1457   }{
1458     \bool_if:NT \g_um_bfuplatin_bool {
1459       \um_map_chars_latin:nn {bfup,bfit} {bfup}
1460       \um_set_mathalphabet_latin:Nnn \mathbf {up,it}{bfup}
1461     }
1462   }
1463 }
1464 \cs_new:Npn \um_config_mathbfup_greek: {
1465   \um_set_mathalphabet_greek:Nnn \mathbfup {up,it}{bfup}
1466   \bool_if:NTF \g_um_bfliteral_bool {
1467     \um_map_chars_greek:nn {bfup}{bfup}
1468     \um_set_mathalphabet_greek:Nnn \mathbf {up}{bfup}
1469   }{
1470     \bool_if:NF \g_um_bfupgreek_bool {
1471       \um_map_chars_greek:nn {bfup,bfit}{bfup}
1472       \um_set_mathalphabet_greek:Nnn \mathbf {up,it}{bfup}
1473     }
1474   }
1475 }
1476 \cs_new:Npn \um_config_mathbfup_greek: {
1477   \um_set_mathalphabet_greek:Nnn \mathbfup {up,it} {bfup}
1478   \bool_if:NTF \g_um_bfliteral_bool {
1479     \um_map_chars_greek:nn {bfup} {bfup}
1480     \um_set_mathalphabet_greek:Nnn \mathbf {up} {bfup}
1481   }{
1482     \bool_if:NT \g_um_bfupgreek_bool {
1483       \um_map_chars_greek:nn {bfup,bfit} {bfup}
1484     }
1485     \bool_if:NT \g_um_bfupgreek_bool {
1486       \um_set_mathalphabet_greek:Nnn \mathbf {up,it} {bfup}
1487     }
1488   }
1489 }
1490 \cs_new:Npn \um_config_mathbfup_misc: {
1491   \um_set_mathalphabet_pos:Nnnn \mathbfup {partial} {up,it}{bfup}
1492   \um_set_mathalphabet_pos:Nnnn \mathbfup {Nabla} {up,it}{bfup}
1493   \um_set_mathalphabet_pos:Nnnn \mathbfup {digamma} {up}{bfup}
1494   \um_set_mathalphabet_pos:Nnnn \mathbfup {Digamma} {up}{bfup}

```

```

1495 \um_set_mathalphabet_pos:Nnnn \mathbf {digamma} {up}{bfup}
1496 \um_set_mathalphabet_pos:Nnnn \mathbf {Digamma} {up}{bfup}
1497 \bool_if:NTF \g_um_bfliteral_bool {
1498   \um_set_mathalphabet_pos:Nnnn \mathbf {partial} {up}{bfup}
1499   \um_set_mathalphabet_pos:Nnnn \mathbf {Nabla} {up}{bfup}
1500 }{
1501   \bool_if:NT \g_um_upNabla_bool {
1502     \um_set_mathalphabet_pos:Nnnn \mathbf {Nabla} {up,it}{bfup}
1503   }
1504   \bool_if:NT \g_um_uppartial_bool {
1505     \um_set_mathalphabet_pos:Nnnn \mathbf {partial} {up,it}{bfup}
1506   }
1507 }
1508 }

```

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

```

1509 \cs_new:Npn \um_config_mathbffrak_Latin: {
1510   \um_set_mathalphabet_Latin:Nnn \mathbffrak {up,it}{bffrak}
1511 }
1512 \cs_new:Npn \um_config_mathbffrak_latin: {
1513   \um_set_mathalphabet_latin:Nnn \mathbffrak {up,it}{bffrak}
1514 }

```

9.1.12 Bold script or calligraphic: `\mathbfscr`

```

1515 \cs_new:Npn \um_config_mathbfscr_Latin: {
1516   \um_set_mathalphabet_Latin:Nnn \mathbfscr {up,it}{bfscr}
1517 }
1518 \cs_new:Npn \um_config_mathbfscr_latin: {
1519   \um_set_mathalphabet_latin:Nnn \mathbfscr {up,it}{bfscr}
1520 }

```

9.1.13 Bold upright sans serif: `\mathbfsfup`

```

1521 \cs_new:Npn \um_config_mathbfsfup_num: {
1522   \um_set_mathalphabet_numbers:Nnn \mathbfsf {up}{bfsfup}
1523   \um_set_mathalphabet_numbers:Nnn \mathbfsfup {up}{bfsfup}
1524 }
1525 \cs_new:Npn \um_config_mathbfsfup_Latin: {
1526   \bool_if:NTF \g_um_sfliteral_bool {
1527     \um_map_chars_Latin:nn {bfsfup} {bfsfup}
1528     \um_set_mathalphabet_Latin:Nnn \mathbfsf {up}{bfsfup}
1529   }{
1530     \bool_if:NT \g_um_upsans_bool {
1531       \um_map_chars_Latin:nn {bfsfup,bfsfit} {bfsfup}
1532       \um_set_mathalphabet_Latin:Nnn \mathbfsf {up,it}{bfsfup}
1533     }
1534   }

```

```

1535 \um_set_mathalphabet_Latin:Nnn \mathbfsfup {up,it}{bfsfup}
1536 }
1537 \cs_new:Npn \um_config_mathbfsfup_latin: {
1538   \bool_if:NTF \g_um_sfliteral_bool {
1539     \um_map_chars_latin:nn {bfsfup} {bfsfup}
1540     \um_set_mathalphabet_latin:Nnn \mathbfsf {up}{bfsfup}
1541   }{
1542     \bool_if:NT \g_um_upsans_bool {
1543       \um_map_chars_latin:nn {bfsfup,bfsfit} {bfsfup}
1544       \um_set_mathalphabet_latin:Nnn \mathbfsf {up,it}{bfsfup}
1545     }
1546   }
1547   \um_set_mathalphabet_latin:Nnn \mathbfsfup {up,it}{bfsfup}
1548 }
1549 \cs_new:Npn \um_config_mathbfsfup_Greek: {
1550   \bool_if:NTF \g_um_sfliteral_bool {
1551     \um_map_chars_Greek:nn {bfsfup}{bfsfup}
1552     \um_set_mathalphabet_Greek:Nnn \mathbfsf {up}{bfsfup}
1553   }{
1554     \bool_if:NT \g_um_upsans_bool {
1555       \um_map_chars_Greek:nn {bfsfup,bfsfit}{bfsfup}
1556       \um_set_mathalphabet_Greek:Nnn \mathbfsf {up,it}{bfsfup}
1557     }
1558   }
1559   \um_set_mathalphabet_Greek:Nnn \mathbfsfup {up,it}{bfsfup}
1560 }
1561 \cs_new:Npn \um_config_mathbfsfup_greek: {
1562   \bool_if:NTF \g_um_sfliteral_bool {
1563     \um_map_chars_greek:nn {bfsfup} {bfsfup}
1564     \um_set_mathalphabet_greek:Nnn \mathbfsf {up} {bfsfup}
1565   }{
1566     \bool_if:NT \g_um_upsans_bool {
1567       \um_map_chars_greek:nn {bfsfup,bfsfit} {bfsfup}
1568       \um_set_mathalphabet_greek:Nnn \mathbfsf {up,it} {bfsfup}
1569     }
1570   }
1571   \um_set_mathalphabet_greek:Nnn \mathbfsfup {up,it} {bfsfup}
1572 }
1573 \cs_new:Npn \um_config_mathbfsfup_misc: {
1574   \um_set_mathalphabet_pos:Nnnn \mathbfsfup {partial} {up,it}{bfsfup}
1575   \um_set_mathalphabet_pos:Nnnn \mathbfsfup {Nabla} {up,it}{bfsfup}
1576   \bool_if:NTF \g_um_sfliteral_bool {
1577     \um_set_mathalphabet_pos:Nnnn \mathbfsf {partial} {up}{bfsfup}
1578     \um_set_mathalphabet_pos:Nnnn \mathbfsf {Nabla} {up}{bfsfup}
1579   }{
1580     \bool_if:NT \g_um_upNabla_bool {

```



```

1581     \um_set_mathalphabet_pos:Nnnn \mathbfsf {Nabla} {up,it}{bfsfup}
1582   }
1583   \bool_if:NT \g_um_uppartial_bool {
1584     \um_set_mathalphabet_pos:Nnnn \mathbfsf {partial} {up,it}{bfsfup}
1585   }
1586 }
1587 }

```

9.1.14 Bold italic sans serif: `\mathbfsfit`

```

1588 \cs_new:Npn \um_config_mathbfsfit_Latin: {
1589   \bool_if:NTF \g_um_sfliteral_bool {
1590     \um_map_chars_Latin:nn {bfsfit} {bfsfit}
1591     \um_set_mathalphabet_Latin:Nnn \mathbfsf {it}{bfsfit}
1592   }{
1593     \bool_if:NF \g_um_upsans_bool {
1594       \um_map_chars_Latin:nn {bfsfup,bfsfit} {bfsfit}
1595       \um_set_mathalphabet_Latin:Nnn \mathbfsf {up,it}{bfsfit}
1596     }
1597   }
1598   \um_set_mathalphabet_Latin:Nnn \mathbfsfit {up,it}{bfsfit}
1599 }
1600 \cs_new:Npn \um_config_mathbfsfit_latin: {
1601   \bool_if:NTF \g_um_sfliteral_bool {
1602     \um_map_chars_latin:nn {bfsfit} {bfsfit}
1603     \um_set_mathalphabet_latin:Nnn \mathbfsf {it}{bfsfit}
1604   }{
1605     \bool_if:NF \g_um_upsans_bool {
1606       \um_map_chars_latin:nn {bfsfup,bfsfit} {bfsfit}
1607       \um_set_mathalphabet_latin:Nnn \mathbfsf {up,it}{bfsfit}
1608     }
1609   }
1610   \um_set_mathalphabet_latin:Nnn \mathbfsfit {up,it}{bfsfit}
1611 }
1612 \cs_new:Npn \um_config_mathbfsfit_Greek: {
1613   \bool_if:NTF \g_um_sfliteral_bool {
1614     \um_map_chars_Greek:nn {bfsfit}{bfsfit}
1615     \um_set_mathalphabet_Greek:Nnn \mathbfsf {it}{bfsfit}
1616   }{
1617     \bool_if:NF \g_um_upsans_bool {
1618       \um_map_chars_Greek:nn {bfsfup,bfsfit}{bfsfit}
1619       \um_set_mathalphabet_Greek:Nnn \mathbfsf {up,it}{bfsfit}
1620     }
1621   }
1622   \um_set_mathalphabet_Greek:Nnn \mathbfsfit {up,it}{bfsfit}
1623 }
1624 \cs_new:Npn \um_config_mathbfsfit_greek: {

```

```

1625 \bool_if:NTF \g_um_sfliteral_bool {
1626   \um_map_chars_greek:nn {bfsfit} {bfsfit}
1627   \um_set_mathalphabet_greek:Nnn \mathbfsf {it} {bfsfit}
1628 }{
1629   \bool_if:NF \g_um_upsans_bool {
1630     \um_map_chars_greek:nn {bfsfup,bfsfit} {bfsfit}
1631     \um_set_mathalphabet_greek:Nnn \mathbfsf {up,it} {bfsfit}
1632   }
1633 }
1634 \um_set_mathalphabet_greek:Nnn \mathbfsfit {up,it} {bfsfit}
1635 }
1636 \cs_new:Npn \um_config_mathbfsfit_misc: {
1637   \um_set_mathalphabet_pos:Nnnn \mathbfsfit {partial} {up,it}{bfsfit}
1638   \um_set_mathalphabet_pos:Nnnn \mathbfsfit {Nabla} {up,it}{bfsfit}
1639   \bool_if:NTF \g_um_sfliteral_bool {
1640     \um_set_mathalphabet_pos:Nnnn \mathbfsf {partial} {it}{bfsfit}
1641     \um_set_mathalphabet_pos:Nnnn \mathbfsf {Nabla} {it}{bfsfit}
1642   }{
1643     \bool_if:NF \g_um_upNabla_bool {
1644       \um_set_mathalphabet_pos:Nnnn \mathbfsf {Nabla} {up,it}{bfsfit}
1645     }
1646     \bool_if:NF \g_um_uppartial_bool {
1647       \um_set_mathalphabet_pos:Nnnn \mathbfsf {partial} {up,it}{bfsfit}
1648     }
1649   }
1650 }

```

10 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 `"ABCDEF` into the \TeX ‘caret input’ form `^^^abc`def. 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.

```

1651 \beginingroup
1652   \char_make_other:N \^
1653   \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1654     \lowercase{
1655       \tl_rescan:nn {
1656         \char_make_other:N \{
1657         \char_make_other:N \}

```

```

1658         \char_make_other:N \&
1659         \char_make_other:N \%
1660         \char_make_other:N \$
1661     }{
1662         \global\let#1=^^^^#2
1663     }
1664 }
1665 }

```

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

```

1666 \gdef\um@scanactivedef"#1\@nil#2{
1667     \lowercase{
1668         \tl_rescan:nn{
1669             \ExplSyntaxOn
1670             \char_make_math_superscript:N\^
1671         }{
1672             \global\def^^^^#1{#2}
1673         }
1674     }
1675 }
1676 \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.

```

1677 \begingroup
1678     \char_make_math_superscript:N\^
1679     \def\UnicodeMathSymbol#1#2#3#4{
1680         \um@scancharlet#2=#1\@nil\ignorespaces
1681     }
1682     \char_make_other:N \#
1683     \@input{unicode-math-table.tex}
1684 \endgroup

```

Fix \backslash:

```

1685 \group_begin:
1686     \lccode`\*=`\
1687     \char_make_escape:N \
1688     \char_make_other:N \
1689     |lowercase{
1690 |group_end:|let|backslash=*}

```

11 Epilogue

Lots of little things to tidy up.

11.0.15 Primes

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

U+2032: PRIME (`\sprime`): x'
U+2033: DOUBLE PRIME (`\dprime`): x''
U+2034: TRIPLE PRIME (`\trprime`): x'''
U+2057: QUADRUPLE PRIME (`\qprime`): 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.

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, `\sprime`, end.
- If pcount=2, check `\dprime`; if it exists, use it, end; if not, goto last step.
- Ditto pcount=3 & `\trprime`.
- Ditto pcount=4 & `\qprime`.
- If pcount>4 or the glyph doesn’t exist, insert pcount `\primes` with `\primekern` between each.

```
1691 \muskip_new:N \g_um_primekern_muskip
1692 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1693 \num_new:N \l_um_primecount_num
1694 \cs_new:Nn \um_nprimes:n {
1695   ^{
1696     \sprime
1697     \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \sprime }
1698   }
1699 }
1700 \cs_new:Nn \um_nprimes_select:n {
```

```

1701 \prg_case_int:nnn {#1}{
1702   {1} { ^{\sprime} }
1703   {2} {
1704     \um_glyph_if_exist:nTF {"2033} { ^{\dprime} } {\um_nprimes:n {#1}}
1705   }
1706   {3} {
1707     \um_glyph_if_exist:nTF {"2034} { ^{\trprime} } {\um_nprimes:n {#1}}
1708   }
1709   {4} {
1710     \um_glyph_if_exist:nTF {"2057} { ^{\qprime} } {\um_nprimes:n {#1}}
1711   }
1712 }{
1713   \um_nprimes:n {#1}
1714 }
1715 }

```

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.

```

1716 \cs_new:Nn \um_scanprime: {
1717   \num_zero:N \l_um_primecount_num
1718   \um_scanprime_collect:
1719 }
1720 \cs_new:Nn \um_scanprime_collect: {
1721   \num_incr:N \l_um_primecount_num
1722   \peek_meaning_remove:NTF ' {
1723     \um_scanprime_collect:
1724   }{
1725     \peek_meaning_remove:NTF \um_scanprime: {
1726       \um_scanprime_collect:
1727     }{
1728       \peek_meaning_remove:NTF ^^^^2032 {
1729         \um_scanprime_collect:
1730       }{
1731         \um_nprimes_select:n {\l_um_primecount_num}
1732       }
1733     }
1734   }
1735 }
1736 \cs_set_eq:NN \prime \um_scanprime:
1737 \group_begin:
1738   \char_make_active:N \'
1739   \char_make_active:n {"2032}
1740   \cs_gset_eq:NN ' \um_scanprime:
1741   \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1742 \group_end:

```

11.0.16 Unicode radicals

Undo the damage made to `\sqrt`:

```
1743 \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`.

```
1744 \def\r@@t#1#2{
1745   \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
1746   \um@scaled@apply{#1}{\kern}{\fontdimen63\l_um_font}
1747   \raise \dimexpr(
1748     \um_fontdimen_to_percent:nn{65}{\l_um_font}\ht\z@-
1749     \um_fontdimen_to_percent:nn{65}{\l_um_font}\dp\z@
1750   )\relax
1751   \copy \rootbox
1752   \um@scaled@apply{#1}{\kern}{\fontdimen64\l_um_font}
1753   \box \z@
1754 }
```

11.0.17 Unicode sub- and super-scripts

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

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

First, the setup of each mathactive char:

```
1755 \prop_new:N \g_um_supers_prop
1756 \prop_new:N \g_um_subs_prop
1757
1758 \group_begin:
1759
1760 % Populate a property list with superscript characters; their mean-
1761 % ing as their key,
1762 % for reasons that will become apparent soon, and their replace-
1763 % ment as each key's value.
1764 % Then make the superscript active and bind it to the scanning function.
1765 %
1766 % \cs{scantokens} makes this process much simpler since we can acti-
1767 % vate the char
```

```

1765 % and assign its meaning in one step.
1766 \cs_set:Nn \um_setup_active_superscript:nn {
1767   \prop_gput:Nxn \g_um_supers_prop   {\meaning #1} {#2}
1768   \char_make_active:n {\`#1}
1769   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1770   \scantokens{
1771     \cs_gset:Npn #1 {
1772       \tl_set:Nn \l_um_ss_chain_tl {#2}
1773       \cs_set_eq:NN \um_sub_or_super:n \sp
1774       \tl_set:Nn \l_um_tmpa_tl {supers}
1775       \um_scan_sscript:
1776     }
1777   }
1778 }
1779
1780 \um_setup_active_superscript:nn {^^^2070} {0}
1781 \um_setup_active_superscript:nn {^^^00b9} {1}
1782 \um_setup_active_superscript:nn {^^^00b2} {2}
1783 \um_setup_active_superscript:nn {^^^00b3} {3}
1784 \um_setup_active_superscript:nn {^^^2074} {4}
1785 \um_setup_active_superscript:nn {^^^2075} {5}
1786 \um_setup_active_superscript:nn {^^^2076} {6}
1787 \um_setup_active_superscript:nn {^^^2077} {7}
1788 \um_setup_active_superscript:nn {^^^2078} {8}
1789 \um_setup_active_superscript:nn {^^^2079} {9}
1790 \um_setup_active_superscript:nn {^^^207a} {+}
1791 \um_setup_active_superscript:nn {^^^207b} {-}
1792 \um_setup_active_superscript:nn {^^^207c} {=}
1793 \um_setup_active_superscript:nn {^^^207d} {(}
1794 \um_setup_active_superscript:nn {^^^207e} {)}
1795 \um_setup_active_superscript:nn {^^^2071} {i}
1796 \um_setup_active_superscript:nn {^^^207f} {n}
1797
1798 % Ditto above.
1799 \cs_set:Nn \um_setup_active_subscript:nn {
1800   \prop_gput:Nxn \g_um_subs_prop   {\meaning #1} {#2}
1801   \char_make_active:n {\`#1}
1802   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1803   \scantokens{
1804     \cs_gset:Npn #1 {
1805       \tl_set:Nn \l_um_ss_chain_tl {#2}
1806       \cs_set_eq:NN \um_sub_or_super:n \sb
1807       \tl_set:Nn \l_um_tmpa_tl {subs}
1808       \um_scan_sscript:
1809     }
1810   }

```

```

1811 }
1812
1813 \um_setup_active_subscript:nn {^^^2080} {0}
1814 \um_setup_active_subscript:nn {^^^2081} {1}
1815 \um_setup_active_subscript:nn {^^^2082} {2}
1816 \um_setup_active_subscript:nn {^^^2083} {3}
1817 \um_setup_active_subscript:nn {^^^2084} {4}
1818 \um_setup_active_subscript:nn {^^^2085} {5}
1819 \um_setup_active_subscript:nn {^^^2086} {6}
1820 \um_setup_active_subscript:nn {^^^2087} {7}
1821 \um_setup_active_subscript:nn {^^^2088} {8}
1822 \um_setup_active_subscript:nn {^^^2089} {9}
1823 \um_setup_active_subscript:nn {^^^208a} {+}
1824 \um_setup_active_subscript:nn {^^^208b} {-}
1825 \um_setup_active_subscript:nn {^^^208c} {=}
1826 \um_setup_active_subscript:nn {^^^208d} {(}
1827 \um_setup_active_subscript:nn {^^^208e} {)}
1828 \um_setup_active_subscript:nn {^^^2090} {a}
1829 \um_setup_active_subscript:nn {^^^2091} {e}
1830 \um_setup_active_subscript:nn {^^^1d62} {i}
1831 \um_setup_active_subscript:nn {^^^2092} {o}
1832 \um_setup_active_subscript:nn {^^^1d63} {r}
1833 \um_setup_active_subscript:nn {^^^1d64} {u}
1834 \um_setup_active_subscript:nn {^^^1d65} {v}
1835 \um_setup_active_subscript:nn {^^^2093} {x}
1836 \um_setup_active_subscript:nn {^^^1d66} {\beta}
1837 \um_setup_active_subscript:nn {^^^1d67} {\gamma}
1838 \um_setup_active_subscript:nn {^^^1d68} {\rho}
1839 \um_setup_active_subscript:nn {^^^1d69} {\phi}
1840 \um_setup_active_subscript:nn {^^^1d6a} {\chi}
1841
1842 \group_end:
1843
1844 % The scanning command, evident in its purpose:
1845 \cs_new:Nn \um_scan_sscript: {
1846   \um_scan_sscript:TF {
1847     \um_scan_sscript:
1848   }{
1849     \um_sub_or_super:n {\l_um_ss_chain_tl}
1850   }
1851 }
1852
1853 % The main theme here is stolen from the source to the vari-
1854 % ous \cs{peek_} functions.
1855 % Consider this function as simply boilerplate:
1856 \cs_new:Nn \um_scan_sscript:TF {

```



```

1856 \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
1857 \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
1858 \tl_set:Nx \l_peek_false_tl {\exp_not:n{\group_align_safe_end: #2}}
1859 \group_align_safe_begin:
1860 \peek_after:NN \um_peek_execute_branches_ss:
1861 }
1862
1863 % We do not skip spaces when scanning ahead, and we explicitly wish to
1864 % bail out on encountering a space or a brace.
1865 \cs_new:Npn \um_peek_execute_branches_ss: {
1866   \bool_if:nTF {
1867     \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
1868     \token_if_eq_catcode_p:NN \l_peek_token \c_group_end_token ||
1869     \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
1870   }
1871   { \l_peek_false_tl }
1872   { \um_peek_execute_branches_ss_aux: }
1873 }
1874
1875 % This is the actual comparison code.
1876 % Because the peeking has already tokenised the next token,
1877 % it's too late to extract its charcode directly. Instead,
1878 % we look at its meaning, which remains a `character' even
1879 % though it is itself math-active. If the character is ever
1880 % made fully active, this will break our assumptions!
1881 %
1882 % If the char's meaning exists as a property list key, we
1883 % build up a chain of sub-/superscripts and iterate. (If not, exit and
1884 % typeset what we've already collected.)
1885 \cs_new:Nn \um_peek_execute_branches_ss_aux: {
1886   \prop_if_in:cxTF
1887     {g_um_\l_um_tmpa_tl _prop}
1888     {\meaning\l_peek_token}
1889   {
1890     \prop_get:cxN
1891       {g_um_\l_um_tmpa_tl _prop}
1892       {\meaning\l_peek_token}
1893     \l_um_tmppb_tl
1894     \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmppb_tl
1895     \l_peek_true_tl
1896   }
1897   {\l_peek_false_tl}
1898 }

```

11.0.18 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`:

```
1899 \def\operator@font{\um_setup_mathup:}
1900 \def\to{\rightarrow}
1901 \def\overrightarrow{\vec}
1902 \def\le{\leq}
1903 \def\ge{\geq}
1904 \def\neq{\neq}
1905 \def\triangle{\mathord{\bigtriangleup}}
1906 \def\bigcirc{\mdlgwhtcircle}
1907 \def\circ{\vysmwhtcircle}
1908 \def\mathyen{\yen}
1909 \def\mathsterling{\sterling}
```

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

```
1910 \@ifpackageloaded{amsmath}{
1911   % define their own colon, perhaps I should just steal it.
1912 }{
1913   \cs_set_protected:Npn \colon {
1914     \bool_if:NTF \g_um_literal_colon_bool {:} { \mathpunct{:} }
1915   }
1916 }
```

`\mathcal`

```
1917 \def\mathcal{\mathscr}
```

`\mathrm`

```
1918 \def\mathrm{\mathup}
1919 \let\mathfence\mathord
```

11.0.19 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:

```
1920 \bool_new:N \g_um_amsmath_bool
1921 \@ifpackageloaded{amsmath}{
1922   \bool_set_true:N \g_um_amsmath_bool
1923 }{
1924   \bool_set_false:N \g_um_amsmath_bool
```

```

1925 }
1926 \bool_if:NT \g_um_amsmath_bool {
1927   \tl_remove_in:Nn \@begindocumenthook {
1928     \mathchardef\std@minus\mathcode`\-\relax
1929     \mathchardef\std@equal\mathcode`\=\relax
1930   }
1931   \AtBeginDocument {
1932     \def\std@minus{\XeTeXmathcharnum\XeTeXmathcodenum`\-\relax}
1933     \def\std@equal{\XeTeXmathcharnum\XeTeXmathcodenum`\=\relax}
1934   }
1935 }

```

- 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:

```

1936 \@ifpackageloaded{amsopn}{
1937   \cs_set:Npn \newmcodes@ {
1938     \mathcode`\'39
1939     \mathcode`\*42
1940     \mathcode`\."613A%
1941     % \ifnum\mathcode`\-=45 \else
1942     %   \mathchardef\std@minus\mathcode`\-\relax
1943     % \fi
1944     \mathcode`\-45
1945     \mathcode`\ /47
1946     \mathcode`\:"603A\relax
1947   }
1948 }{}

```

- `\mathinner` items:

```

1949 \cs_set:Npn \mathellipsis {\mathinner{\unicodeellipsis}}
1950 \cs_set:Npn \cdots {\mathinner{\unicodcdots}}
1951 \bool_if:NT \g_um_amsmath_bool {
1952   \cs_set_eq:NN \@cdots \cdots
1953   \cs_set_eq:NN \dotso \cdots
1954 }

```

Octothorpe is an odd one:

```

1955 \AtBeginDocument{
1956   \def\widehat{\hat}
1957   \def\widetilde{\tilde}
1958 }

```

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

```

\Digamma
1959 \def\digamma{\updigamma}
1960 \def\Digamma{\upDigamma}

```

Overriding amsmath definitions:

```
1961 \AtBeginDocument{
1962   \def\@cdots{\mathinner{\cdots}}
1963 }
```

Interaction with beamer:

```
1964 \@ifclassloaded{beamer}{
1965   \ifbeamer@suppressreplacements\else
1966     \PackageWarningNoLine{unicode-math}{
1967       Disabling~ beamer's~ math~ setup.^{}
1968       Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1969     }
1970     \beamer@suppressreplacementstrue
1971   \fi
1972 }{}
```

The end.

```
1973 \ExplSyntaxOff
```

12 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.

```
1974 #!/bin/sh
1975
1976 cat stix-tbl.txt |
1977 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!)

```
1978 {if (usv != substr($0,2,5) && substr($0,2,1) != " ")
1979   {usv = substr($0,2,5);
1980     texname = substr($0,84,25);
1981     class = substr($0,57,1);
1982     description = tolower(substr($0,233,350));
```

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

```

1983     if (texname      ~ /[\\]/ &&
1984         substr(texname,0,5) != "\\text"    &&
1985         substr(texname,0,4) != "\\ipa"     &&
1986         substr(texname,0,5) != "\\tone"    &&
1987         substr(texname,3,1) != " "        &&
1988         class        != " "              &&
1989         description !~ /<reserved>/ )

```

Print the actual entry corresponding to the unicode character:

```

1990     print "\\UnicodeMathSymbol{\"\" \" \
1991         usv \"}\" \" \
1992         texname \"}\" \" \
1993         class \"}\" \" \
1994         description \"%\";
1995     } }' - |

```

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

```

1996 sed -e ' s/{N}/{\\mathord}/ ' \

```

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

```

1997     -e ' s/{F}/{\\mathord}/ ' \
1998     -e ' s/{A}/{\\mathalpha}/ ' \
1999     -e ' s/{D}/{\\mathaccent}/ ' \
2000     -e ' s/{P}/{\\mathpunct}/ ' \
2001     -e ' s/{B}/{\\mathbin}/ ' \
2002     -e ' s/{R}/{\\mathrel}/ ' \
2003     -e ' s/{L}/{\\mathop}/ ' \
2004     -e ' s/{O}/{\\mathopen}/ ' \
2005     -e ' s/{C}/{\\mathclose}/ ' \

```

Fixing up a couple of things in the STIX table.

```

2006     -e ' s/\\^/\\string^/ ' > unicode-math.tex

```

A Documenting maths support in the NFSS

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

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

`\DeclareSymbolFont{ $\langle name \rangle$ } $\langle NFSS decl. \rangle$`

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

Maths alphabet fonts Fonts for $ABC-xyz$, $\mathfrak{A}\mathfrak{B}\mathfrak{C}-\mathcal{X}\mathcal{Y}\mathcal{Z}$, etc.

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

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

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

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

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

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

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

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

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

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

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

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

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

Accents are not included yet.

Summary For symbols, something like:

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

For characters, something like:

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

B \XeTeX math font dimensions

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

<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.
15	<code>AXISHEIGHT</code>	Axis height of the font.
16	<code>ACCENTBASEHEIGHT</code>	Maximum (ink) height of accent base that does not require raising the accents. Suggested: x-height of the font (<code>os2.sxHeight</code>) plus any possible overshoots.

\fontdimen	Dimension name	Description
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: $\frac{1}{5}$ 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×default rule thickness.
26	SUPERSCRIPBOTTOMMAX-WITHSUBSCRIPT	The maximum level to which the (ink) bottom of superscript can be pushed to increase the gap between superscript and subscript, before subscript starts being moved down. Suggested: $\frac{1}{5}$ x-height.

\fontdimen	Dimension name	Description
27	SPACEAFTERScript	Extra white space to be added after each subscript and superscript. Suggested: 0.5pt for a 12 pt font.
28	UPPERLIMITGAPMIN	Minimum gap between the (ink) bottom of the upper limit, and the (ink) top of the base operator.
29	UPPERLIMITBASELINERISEMIN	Minimum distance between baseline of upper limit and (ink) top of the base operator.
30	LOWERLIMITGAPMIN	Minimum gap between (ink) top of the lower limit, and (ink) bottom of the base operator.
31	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	STACKTOPDISPLAYSTYLESHIFTUP	Standard shift up applied to the top element of a stack in display style.
34	STACKBOTTOMSHIFTDOWN	Standard shift down applied to the bottom element of a stack. Positive for moving in the downward direction.
35	STACKBOTTOMDISPLAYSTYLESHIFTDOWN	Standard shift down applied to the bottom element of a stack in display style. Positive for moving in the downward direction.
36	STACKGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element. Suggested: 3×default rule thickness.
37	STACKDISPLAYSTYLEGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element in display style. Suggested: 7×default rule thickness.
38	STRETCHSTACKTOPSHIFTUP	Standard shift up applied to the top element of the stretch stack.
39	STRETCHSTACKBOTTOMSHIFTDOWN	Standard shift down applied to the bottom element of the stretch stack. Positive for moving in the downward direction.

\fontdimen	Dimension name	Description
40	STRETCHSTACKGAPABOVEMIN	Minimum gap between the ink of the stretched element, and the (ink) bottom of the element above. Suggested: UpperLimitGapMin
41	STRETCHSTACKGAPBELOWMIN	Minimum gap between the ink of the stretched element, and the (ink) top of the element below. Suggested: LowerLimitGapMin.
42	FRACTIONNUMERATORSHIFTUP	Standard shift up applied to the numerator.
43	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
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.

\fontdimen	Dimension name	Description
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.
64	RADICALKERNAFTERDEGREE	Negative kern after the degree of a radical, if such is present. Suggested: -10/18 of em.

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

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		
\"	1990
\#	1682
\\$	1660
\%	1659
\&	1658
\'	1738, 1938
*	585, 1686, 1939
\-	584, 1928, 1932, 1941, 1942, 1944
\.	1940
\/	634, 1945
\:	587, 1946
\::	893
\::f	893
\::n	893
\<	638
\=	1929, 1933
\>	639
\@DeclareMathDelimiter	430
\@DeclareMathSizes	422
\@backslashchar	1100, 1119
\@begindocumenthook	1927
\@ccclvi	450
\@cdots	1952, 1962
\@empty	503, 771, 786, 807, 812
\@ifclassloaded	1964
\@ifnextchar	1743
\@ifpackageloaded	...	1910, 1921, 1936
\@ii	770, 771, 773, 775, 778, 783
\@input	567, 1683
\@marker	783, 802
\@nil	457, 615, 783, 796–799, 838, 1653, 1666, 1680
\@preamblecmds	434
\@sqrt	1743
\@tempa	259, 323, 349, 358, 376, 394, 402, 409, 729, 774, 798, 800, 812
\@tempb	..	259, 260, 323, 324, 349, 350, 358, 359, 376, 377, 394, 395, 402, 403, 409, 410, 729, 730, 801, 802, 807
\@tempswafalse	772
\@tempswatru	776, 779, 804, 809, 814, 819	
\@xDeclareMathDelimiter	431
\@xxDeclareMathDelimiter	430
\	1686, 1688, 1983–1986, 1990, 1996–2006	
\{	1656
\}	1657
\^	1652, 1670, 1678, 2006
\	1687
Numbers		
\0	1023
\9	1023
_	1990–1993, 1996–2005
A		
\A	1020
\a	1019
\addnolimits	<u>695</u>
\addtoversion	423
\alloc@	450
\alpha@elt	424
\alpha@list	424
\AtBeginDocument	.	964, 1931, 1955, 1961
\awint	691
B		
\B	35
\beamer@suppressreplacementstrue	1970	
\begingroup	454, 1651, 1677
\beta	1836
\bgroup	1136
\bigcirc	1906
\bigtriangleup	1905
\bool_if:NF	586, 1198, 1209, 1220, 1230, 1331, 1343, 1360, 1368, 1375, 1383, 1395, 1407, 1410, 1422, 1425, 1470, 1593, 1605, 1617, 1629, 1643, 1646

\bool_if:NT	1153, 1163, 1174, 1184, 1307, 1319, 1435, 1443, 1450, 1458, 1482, 1485, 1501, 1504, 1530, 1542, 1554, 1566, 1580, 1583, 1926, 1951
\bool_if:NTF	135–142, 366, 384, 548, 589, 598, 706, 977, 980, 983, 986, 1150, 1160, 1171, 1181, 1195, 1205, 1217, 1227, 1303, 1315, 1327, 1339, 1364, 1379, 1391, 1403, 1418, 1439, 1454, 1466, 1478, 1497, 1526, 1538, 1550, 1562, 1576, 1589, 1601, 1613, 1625, 1639, 1914
\bool_if:nTF	1866
\bool_new:N ..	13–28, 346, 347, 401, 1920
\bool_set_false:N 261–273, 276–278, 280, 284, 286, 287, 292, 300, 301, 314, 315, 320, 325–329, 331, 333, 337, 351, 362, 380, 396, 406, 536, 541, 734, 1924
\bool_set_true:N	275, 279, 281–283, 285, 289–291, 293–299, 303–313, 317–319, 332, 334, 335, 338–341, 343, 353, 355, 360, 378, 398, 404, 501, 534, 539, 731, 1922
\box	1753
C	
\C	36
\c_group_begin_token	1867
\c_group_end_token	1868
\c_peek_true_remove_next_tl ...	1857
\c_space_token	1869
\cdots	1950, 1952, 1953, 1962
\cdp@elt	421
\cdp@list	421
\char_make_active:N	1738
\char_make_active:n	455, 1739, 1768, 1801
\char_make_escape:N	1687
\char_make_math_superscript:N	1670, 1678
\char_make_other:N 1652, 1656–1660, 1682, 1688
\chardef	450
\chi	1840
\circ	1907
\cirfnint	691
\clist_map_break:	1092
\clist_map_inline:Nn	682
\clist_map_inline:nn 620, 675, 737, 827, 852, 857, 862, 873, 885, 922, 928, 933, 938, 944, 950, 966, 1089, 1095
\clist_map_variable:nNn	895, 90
\colon	1913
\copy	1751
\cs	1764, 1853
\cs_generate_variant:Nn ...	7–11, 889–892, 960–963, 1106, 1126–1129
\cs_gset:cpn	468, 483, 484
\cs_gset:cpx	461
\cs_gset:Npn ..	470, 476, 1653, 1771, 1804
\cs_gset:Npx	487
\cs_gset_eq:NN	1740, 1741
\cs_if_exist:cF	1131
\cs_if_exist:cT	921, 1114
\cs_new:Nn	29, 32, 583, 614, 619, 624, 627, 633, 668, 671, 833, 837, 894, 900, 910, 913, 916, 920, 927, 932, 937, 943, 949, 965, 1031, 1086, 1113, 1130, 1694, 1700, 1716, 1720, 1845, 1855, 1885
\cs_new:Npn .	1145, 1149, 1159, 1170, 1180, 1190, 1194, 1204, 1216, 1226, 1236, 1240, 1243, 1253, 1256, 1263, 1270, 1281, 1287, 1295, 1298, 1302, 1314, 1326, 1338, 1350, 1353, 1356, 1359, 1374, 1389, 1401, 1415, 1430, 1434, 1449, 1464, 1476, 1490, 1509, 1512, 1515, 1518, 1521, 1525, 1537, 1549, 1561, 1573, 1588, 1600, 1612, 1624, 1636, 1865
\cs_set:cpn	1132
\cs_set:Nn	365, 383, 452, 496, 575, 578, 674, 681, 826, 842, 845, 848, 851, 856, 861, 872, 878, 881, 884, 1118, 1766, 1799
\cs_set:Npn	761, 765, 893, 1937, 1949, 1950
\cs_set:Npx	512, 516
\cs_set_eq:cN	1121

<code>\cs_set_eq:NN</code>	551–555, 559–563, 1070, 1079, 1736, 1773, 1806, 1952, 1953
<code>\cs_set_protected:cpn</code>	1135
<code>\cs_set_protected:Npn</code>	1913
<code>\cs_to_str:N</code>	457, 460–462, 468, 483, 484, 676, 1080, 1087
<code>\csname</code>	457, 460, 497, 504, 629, 669, 672, 1139
D	
<code>\D</code>	37
<code>\d</code>	65
<code>\DeclareDocumentCommand</code>	255, 499, 695, 698
<code>\DeclareMathAccent</code>	429
<code>\DeclareMathAlphabet</code>	428
<code>\DeclareMathDelimiter</code>	430
<code>\DeclareMathRadical</code>	431
<code>\DeclareMathSizes</code>	421
<code>\DeclareMathSymbol</code>	429
<code>\DeclareMathVersion</code>	423, 506
<code>\DeclareRobustCommand</code>	1743
<code>\DeclareSymbolFont</code>	426, 565
<code>\DeclareSymbolFontAlphabet</code>	432
<code>\DeclareSymbolFontAlphabet@</code>	432
<code>\def</code>	436, 439, 450, 703, 796, 798–801, 1672, 1679, 1744, 1899–1909, 1917, 1918, 1932, 1933, 1956, 1957, 1959, 1960, 1962
<code>\define@choicekey</code>	258, 323, 348, 358, 376, 394, 402, 409, 729
<code>\define@cmdkey</code>	722–725
<code>\define@key</code>	705
<code>\define@mathalphabet</code>	423
<code>\define@mathgroup</code>	423
<code>\Digamma</code>	1959
<code>\digamma</code>	1959
<code>\dimexpr</code>	437, 538, 1747
<code>\do</code>	434
<code>\dorestore@version</code>	425
<code>\dotsb@</code>	1953
<code>\dp</code>	1749
<code>\dprime</code>	1704
E	
<code>\E</code>	38
<code>\e</code>	66
<code>\edef</code>	797, 798
<code>\egroup</code>	1133, 1138
<code>\else</code>	442, 445, 465, 474, 479, 485, 488, 540, 777, 782, 788, 806, 811, 816, 1941, 1965
<code>\else:</code>	1124
<code>\encodingdefault</code>	566
<code>\endcsname</code>	457, 460, 497, 504, 629, 669, 672, 1139
<code>\endgroup</code>	458, 1676, 1684
<code>\epsilon</code>	976
<code>\errorcontextlines</code>	902
<code>\etex_iffontchar:D</code>	1124
<code>\ExecuteOptionsX</code>	418
<code>\exp_after:wN</code>	750, 753, 1080, 1088
<code>\exp_args:Nc</code>	576
<code>\exp_args:Nnff</code>	893, 896, 904
<code>\exp_args:NV</code>	1091
<code>\exp_not:c</code>	462, 974
<code>\exp_not:N</code>	513, 517, 677
<code>\exp_not:n</code>	463, 1856, 1858
<code>\exp_not:V</code>	513, 739
<code>\expandafter</code>	459, 773, 775, 778, 783, 797, 798, 802, 1138, 1139
<code>\ExplSyntaxOff</code>	1973
<code>\ExplSyntaxOn</code>	6, 1669
F	
<code>\F</code>	39
<code>\f@size</code>	504
<code>\fi</code>	321, 344, 356, 363, 381, 399, 407, 416, 447, 448, 490–494, 547, 685, 732, 780, 781, 784, 790, 792, 793, 805, 810, 815, 820–824, 1943, 1971
<code>\fi:</code>	1124
<code>\fint</code>	691
<code>\font</code>	537
<code>\fontdimen</code>	437, 538, 1746, 1752
<code>\fontname</code>	1063
G	
<code>\g</code>	67
<code>\g@addto@macro</code>	787, 789
<code>\g_um_amsmath_bool</code>	1920, 1922, 1924, 1926, 1951
<code>\g_um_bfit_Greek_usv</code>	141
<code>\g_um_bfit_greek_usv</code>	142

\g_um_bfit_Latin_usv	139	\g_um_fam_int	12, 557, 558
\g_um_bfit_latin_usv	140	\g_um_it_h_usv	1025
\g_um_bfit_Nabla_usv	372, 600, 608	\g_um_it_Nabla_usv	371, 591, 595
\g_um_bfit_partial_usv ..	390, 604, 610	\g_um_it_partial_usv	389, 593, 596
\g_um_bfliteral_bool		\g_um_literal_bool ..	16, 273, 287,
..... 21, 318, 325, 331, 337,		301, 315, 317, 589, 1150, 1160,	
343, 598, 1364, 1379, 1391, 1403,		1171, 1181, 1195, 1205, 1217, 1227	
1418, 1439, 1454, 1466, 1478, 1497		\g_um_literal_colon_bool	
\g_um_bfNabla_up_or_it_usv 401, 404, 406, 586, 1914	
..... 368, 372, 608		\g_um_math_alphabet_name_Greek_tl	
\g_um_bfpartial_up_or_it_usv 1111	
..... 386, 390, 610		\g_um_math_alphabet_name_greek_tl	
\g_um_bfsfit_Greek_usv	137 1110	
\g_um_bfsfit_greek_usv	138	\g_um_math_alphabet_name_Latin_tl	
\g_um_bfsfit_Latin_usv	135 1109	
\g_um_bfsfit_latin_usv	136	\g_um_math_alphabet_name_latin_tl	
\g_um_bfsfit_Nabla_usv ..	373, 602, 609 1108	
\g_um_bfsfit_partial_usv	391, 606, 611	\g_um_math_alphabet_name_num_tl	1112
\g_um_bfsfNabla_up_or_it_usv		\g_um_mathalph_seq	726, 755, <u>989</u>
..... 369, 373, 609		\g_um_mathbb_alph_clist ...	1007, 1036
\g_um_bfsfpartial_up_or_it_usv ..		\g_um_mathbf_alph_clist ...	1012, 1043
..... 387, 391, 611		\g_um_mathbffrak_alph_clist	1006, 1047
\g_um_bfsfup_Greek_usv	137	\g_um_mathbfit_alph_clist	1014, 1045
\g_um_bfsfup_greek_usv	138	\g_um_mathbfscr_alph_clist	1005, 1046
\g_um_bfsfup_Latin_usv	135	\g_um_mathbfssf_alph_clist	1015, 1048
\g_um_bfsfup_latin_usv	136	\g_um_mathbfssfit_alph_clist	1017, 1050
\g_um_bfsfup_Nabla_usv ..	369, 601, 609	\g_um_mathbfssfup_alph_clist	1016, 1049
\g_um_bfsfup_partial_usv	387, 605, 611	\g_um_mathbfup_alph_clist	1013, 1044
\g_um_bfup_Greek_usv	141	\g_um_mathfrak_alph_clist	1004, 1038
\g_um_bfup_greek_usv	142	\g_um_mathit_alph_clist ...	1002, 1035
\g_um_bfup_Latin_usv	139	\g_um_mathit_Greek_usv	1028
\g_um_bfup_latin_usv	140	\g_um_mathit_greek_usv	1027
\g_um_bfup_Nabla_usv	368, 599, 608	\g_um_mathit_Latin_usv	1026
\g_um_bfup_partial_usv ..	386, 603, 610	\g_um_mathit_latin_usv	1025
\g_um_bfupGreek_bool		\g_um_mathscr_alph_clist ..	1003, 1037
..... 24, 141, 265, 279,		\g_um_mathsf_alph_clist ...	1009, 1039
293, 307, 326, 332, 338, 1395, 1470		\g_um_mathsfitt_alph_clist	1011, 1041
\g_um_bfupgreek_bool		\g_um_mathsfup_alph_clist	1010, 1040
.... 25, 142, 266, 280, 294, 308,		\g_um_mathtt_alph_clist ...	1008, 1042
327, 333, 339, 1407, 1410, 1482, 1485		\g_um_mathup_alph_clist ...	1001, 1034
\g_um_bfupLatin_bool		\g_um_mathup_Greek_usv	1022
.... 22, 139, 267, 281, 295, 309,		\g_um_mathup_greek_usv	1021
328, 334, 340, 1360, 1368, 1435, 1443		\g_um_mathup_Latin_usv	1020
\g_um_bfuplatin_bool		\g_um_mathup_latin_usv	1019
.... 23, 140, 268, 282, 296, 310,		\g_um_mathup_num_usv	1023
329, 335, 341, 1375, 1383, 1450, 1458		\g_um_Nabla_up_or_it_usv	367, 371, 595

<code>\g_um_partial_up_or_it_usv</code>	<code>\group_end:</code>
..... 385, 389, 596	1742, 1842
<code>\g_um_primekern_muskip</code> 1691, 1692, 1697	
<code>\g_um_sfliteral_bool</code>	H
. 319, 347, 355, 1303, 1315, 1327,	<code>\H</code>
1339, 1526, 1538, 1550, 1562,	<code>\hat</code>
1576, 1589, 1601, 1613, 1625, 1639	<code>\hbox</code>
<code>\g_um_slash_delimiter_usv</code>	<code>\ht</code>
..... 411, 413, 415, 634–636	
<code>\g_um_subs_prop</code>	I
..... 1756, 1800	<code>\I</code>
<code>\g_um_supers_prop</code>	<code>\i</code>
..... 1755, 1767	<code>\if@tempswa</code>
<code>\g_um_texgreek_bool</code> 28, 272, 286, 300,	<code>\ifbeamer@suppressreplacements</code> .
314, 320, 396, 398, 977, 980, 983, 986	1965
<code>\g_um_up_Nabla_usv</code>	<code>\ifcase</code>
..... 367, 590, 595	260,
<code>\g_um_up_partial_usv</code>	324, 350, 359, 377, 395, 403, 410, 730
..... 385, 592, 596	<code>\ifdim</code>
<code>\g_um_upGreek_bool</code> 538
..... 19, 137, 261, 275, 289, 303, 1174, 1220	<code>\ifnum</code> ..
<code>\g_um_upgreek_bool</code> 683, 803, 808, 813, 817, 818, 1941
..... 20, 138, 262, 276, 290, 304, 1184, 1230	<code>\ifx</code> .
<code>\g_um_upLatin_bool</code> 440, 443, 453, 466, 475, 480, 486,
..... 17, 135, 263, 277, 291, 305, 1153, 1198	771, 774, 775, 778, 786, 802, 807, 812
<code>\g_um_uplatin_bool</code>	<code>\ignorespaces</code>
..... 18, 136, 264, 278, 292, 306, 1163, 1209 1680
<code>\g_um_upNabla_bool</code>	<code>\iiiint</code>
..... 26, 269, 283, 297, 311, 689
360, 362, 366, 1422, 1501, 1580, 1643	<code>\iiint</code>
<code>\g_um_uppartial_bool</code> 689
..... 27, 270, 284, 298, 312,	<code>\iint</code>
378, 380, 384, 1425, 1504, 1583, 1646 689
<code>\g_um_upsans_bool</code>	<code>\init@restore@version</code>
..... 271, 425
285, 299, 313, 346, 351, 353, 1307,	<code>\int</code>
1319, 1331, 1343, 1530, 1542, 689
1554, 1566, 1593, 1605, 1617, 1629	<code>\int_incr:N</code>
<code>\gamma</code> 557
..... 1837	<code>\int_new:N</code>
<code>\gdef</code> 12
..... 1666	<code>\int_use:N</code>
<code>\ge</code> 558
..... 1903	<code>\intBar</code>
<code>\geq</code> 691
..... 1903	<code>\intbar</code>
<code>\get@cdp</code> 691
..... 427	<code>\intcap</code>
<code>\glb@currsize</code> 693
..... 500	<code>\intclockwise</code>
<code>\global</code> 690
456, 459, 471, 472, 477, 478,	<code>\intcup</code>
481, 482, 489, 1662, 1672, 1769, 1802 693
<code>\group@elt</code>	<code>\intlarhk</code>
..... 426 692
<code>\group@list</code>	<code>\intx</code>
..... 426 692
<code>\group_align_safe_begin:</code>	
..... 1859	J
<code>\group_align_safe_end:</code>	<code>\j</code>
..... 1858 70
<code>\group_begin:</code>	K
..... 1685, 1737, 1758	<code>\kern</code>
 1746, 1752
	L
	<code>\L</code>
 42
	<code>\l_peek_false_tl</code>
 1858, 1871, 1897
	<code>\l_peek_token</code> ...
 1867–1869, 1888, 1892
	<code>\l_peek_true_aux_tl</code>
 1856

<code>\l_peek_true_tl</code>	1857, 1895	<code>\mathbb</code>	992, 1036,
<code>\l_um_char_range_seq</code>			1241, 1244–1251, 1254, 1257–1261
 502, 728, 735, 741, 770	<code>\mathbbbit</code>	1264–1268
<code>\l_um_font</code>	441, 444, 537, 538,	<code>\mathbf</code>	
	1063, 1124, 1746, 1748, 1749, 1752		994, 1043, 1366, 1370, 1381, 1385,
<code>\l_um_fontspec_feature_bool</code>			1393, 1397, 1419, 1420, 1423,
 13, 534, 536, 706		1426, 1431, 1441, 1445, 1456,
<code>\l_um_inc_num</code>	903, 905, 906		1460, 1468, 1472, 1480, 1486,
<code>\l_um_init_bool</code> ..	15, 501, 548, 731, 734		1495, 1496, 1498, 1499, 1502, 1505
<code>\l_um_input_num</code>	895, 897, 901, 905	<code>\mathbffrac</code>	995, 1047, 1510, 1513
<code>\l_um_mathalph_seq</code>		<code>\mathbfrit</code>	994, 1045, 1363, 1378,
 727, 736, 739, 1033, 1071		1390, 1402, 1405, 1411, 1416, 1417
<code>\l_um_missing_alph_seq</code>		<code>\mathbfscr</code>	995, 1046, 1516, 1519
 1030, 1032, 1060, 1065, 1099	<code>\mathbfssf</code>	
<code>\l_um_mversion_tf</code>	505, 506		996, 1048, 1522, 1528, 1532, 1540,
<code>\l_um_nolimits_tl</code> ...	463, 688, 696, 699		1544, 1552, 1556, 1564, 1568,
<code>\l_um_ot_math_bool</code>	14, 539, 541		1577, 1578, 1581, 1584, 1591,
<code>\l_um_primecount_num</code>			1595, 1603, 1607, 1615, 1619,
 1693, 1717, 1721, 1731		1627, 1631, 1640, 1641, 1644, 1647
<code>\l_um_radicals_tl</code>	467, 701	<code>\mathbfssfit</code>	996, 1050,
<code>\l_um_script_features_tl</code> ...	507, 524		1598, 1610, 1622, 1634, 1637, 1638
<code>\l_um_script_font_tl</code>	509, 523	<code>\mathbfssfup</code>	996, 1049, 1523,
<code>\l_um_ss_chain_tl</code> 1772, 1805, 1849, 1894			1535, 1547, 1559, 1571, 1574, 1575
<code>\l_um_sscript_features_tl</code> ..	508, 528	<code>\mathbfsup</code>	994, 1044, 1432,
<code>\l_um_sscript_font_tl</code>	510, 527		1438, 1453, 1465, 1477, 1491–1494
<code>\l_um_tmpa_tl</code>	739,	<code>\mathbin</code>	584, 585
	746, 749, 750, 752, 753, 755, 762,	<code>\mathcal</code>	<u>1917</u>
	766, 1072, 1080, 1082, 1087, 1088,	<code>\mathchar@type</code>	432, 460, 470, 472, 476,
	1097, 1101, 1774, 1807, 1887, 1891		478, 481, 483, 484, 487, 489, 497, 628
<code>\l_um_tmpb_tl</code>	739,	<code>\mathchardef</code>	1928, 1929, 1942
	747, 767, 1073, 1078, 1080, 1082,	<code>\mathclose</code>	475, 484
	1088, 1090, 1091, 1096, 1893, 1894	<code>\mathcode</code>	456,
<code>\l_um_tmpe_tl</code> ..	739, 748, 763, 1074, 1075		1928, 1929, 1938–1942, 1944–1946
<code>\lccode</code>	1686	<code>\mathellipsis</code>	1949
<code>\le</code>	1902	<code>\mathfence</code>	480, 1919
<code>\left</code>	<u>702</u>	<code>\mathfrak</code> ...	992, 1038, 1288–1293, 1296
<code>\left@primitive</code>	702, 703	<code>\mathgroup</code>	450
<code>\leq</code>	1902	<code>\mathinner</code>	1949, 1950, 1962
<code>\let</code>	451, 500, 503, 702, 1662, 1919	<code>\mathit</code>	991, 1035,
<code>\lowercase</code>	1654, 1667		1202, 1214, 1224, 1234, 1237, 1238
<code>\lowint</code>	693	<code>\mathop</code>	453
M		<code>\mathopen</code>	466, 483, 703
<code>\M</code>	43	<code>\mathord</code>	590–593, 595, 596,
<code>\m@th</code>	1745		599–606, 608–611, 625, 1905, 1919
<code>\mathaccent</code>	486	<code>\mathpunct</code>	1914
<code>\mathalpha</code>	677, 835, 838	<code>\mathrel</code>	587

<code>\mathrm</code>	1918
<code>\mathscr</code>	992,
1037, 1271–1279, 1282–1285, 1917	
<code>\mathsf</code> ..	993, 1039, 1299, 1305, 1309,
1317, 1321, 1329, 1333, 1341, 1345	
<code>\mathsf{fit}</code>	993, 1041, 1336, 1348
<code>\mathsf{fup}</code> ...	993, 1040, 1300, 1312, 1324
<code>\mathsterling</code>	1909
<code>\mathhtt</code>	992, 1042, 1351, 1354, 1357
<code>\mathup</code>	991, 1034, 1147, 1157,
1168, 1178, 1188, 1191, 1192, 1918	
<code>\mathyen</code>	1908
<code>\mddefault</code>	566
<code>\mdlgwhtcircle</code>	1906
<code>\meaning</code>	1767, 1800, 1888, 1892
<code>\mitepsilon</code>	977, 983
<code>\mitphi</code>	980, 986
<code>\mitvarepsilon</code>	977, 983
<code>\mitvarphi</code>	980, 986
<code>\mode_if_math:F</code>	1137
<code>\mskip</code>	1697
<code>\muskip_gset:Nn</code>	1692
<code>\muskip_new:N</code>	1691
N	
<code>\N</code>	44
<code>\ne</code>	1904
<code>\neq</code>	1904
<code>\new@mathalphabet</code>	428
<code>\new@mathgroup</code>	421, 450, 451
<code>\new@mathversion</code>	426
<code>\new@symbolfont</code>	427
<code>\newcommand</code>	704, 769
<code>\newfam</code>	451
<code>\newmathalphabet</code>	422
<code>\newmathalphabet@@</code>	422
<code>\newmathalphabet@@@</code>	422
<code>\newmcodes@</code>	1937
<code>\nolimits</code>	463
<code>\non@alpherr</code>	1139
<code>\npolint</code>	692
<code>\num_incr:N</code>	1721
<code>\num_new:N</code>	1693
<code>\num_zero:N</code>	1717
<code>\number</code>	897, 905, 906
<code>\numexpr</code>	808, 813, 817, 818, 835, 897, 905, 906

O	
<code>\o</code>	71
<code>\oiint</code>	689
<code>\oiiint</code>	689
<code>\oint</code>	689
<code>\ointctrclockwise</code>	690
<code>\operator@font</code>	1899
<code>\or</code> ..	274, 288, 302, 316, 330, 336, 342,
352, 354, 361, 379, 397, 405, 412, 414	
<code>\overrightarrow</code>	1901

P	
<code>\P</code>	45
<code>\PackageError</code>	709
<code>\PackageInfo</code>	550
<code>\PackageWarning</code>	1076
<code>\PackageWarningNoLine</code>	542, 1966
<code>\peek_after:NN</code>	1860
<code>\peek_meaning_remove:NTF</code>	1722, 1725, 1728
<code>\phi</code>	979, 1839
<code>\pointint</code>	692
<code>\prg_case_int:nnn</code>	1701
<code>\prg_do_nothing:</code>	1121
<code>\prg_new_conditional:Nnn</code> ..	745, 1123
<code>\prg_replicate:nn</code>	1697
<code>\prg_return_false:</code>	758, 1124
<code>\prg_return_true:</code>	756, 1124
<code>\prg_stepwise_inline:nnnn</code>	828
<code>\prg_stepwise_variable:nnnNn</code>	903
<code>\prime</code>	1736
<code>\process@table</code>	425
<code>\ProcessOptionsX</code>	419
<code>\prop_get:cxN</code>	1890
<code>\prop_get:NnN</code>	10
<code>\prop_gput:Nnn</code>	9
<code>\prop_gput:Nxn</code>	1767, 1800
<code>\prop_if_in:cxTF</code>	1886
<code>\prop_if_in:NnTF</code>	11
<code>\prop_new:N</code>	1755, 1756
<code>\protect</code>	712
<code>\ProvidesPackage</code>	1

Q	
<code>\Q</code>	46
<code>\q_nil</code>	750, 753, 761, 765
<code>\qprime</code>	1710

R	
\R	47
\r@@t	1744
\raise	1747
\relax	260, 324, 350, 437, 453, 456, 460, 466, 468, 470–472, 475–478, 481–484, 486, 487, 489, 497, 500, 537, 538, 683, 730, 775, 778, 802, 803, 808, 813, 817, 818, 835, 897, 905, 906, 1750, 1928, 1929, 1932, 1933, 1942, 1946
\removenolimits	698
\RequirePackage	3–5
\restore@mathversion	425
\rho	1838
\rightarrow	1900
\rootbox	1751
\rppolint	691
S	
\sb	1806
\scan_stop: ..	630, 631, 1124, 1769, 1802
\scantokens	1770, 1803
\scpolint	692
\scriptscriptstyle	443
\scriptstyle	440
\seq_clear:N ...	502, 735, 736, 989, 1032
\seq_if_empty:N	1060
\seq_if_empty:N	1033
\seq_if_in:Nn	8
\seq_if_in:N	755
\seq_map_inline:Nn	1065, 1071
\seq_map_variable:NNn	770
\seq_new:N	726–728, 1030
\seq_put_right:Nn	741, 998
\seq_put_right:Nx	739, 1099
\set@@mathdelimiter	431
\set@mathaccent	429
\set@mathchar	429
\set@mathdelimiter	431
\set@mathsymbol	430
\setbox	1745
\setkeys	256, 511, 513
\SetMathAlphabet	428
\SetMathAlphabet@	428
\setmathfont	499, 712
\SetSymbolFont	427
\SetSymbolFont@	427
\sf@size	522, 526
\sp	1773
\space	1066, 1101, 1139
\sprime	625, 1696, 1697, 1702
\sqint	692
\sqrt	701, 1743
\sqrtsign	1743, 1745
\std@equal	1929, 1933
\std@minus	1928, 1932, 1942
\sterling	1909
\string	797, 798
\strip@pt	437
\sumint	690
T	
\tf@size	521, 522
\thinmuskip	1692
\tilde	1957
\tl_if_empty:N	1075
\tl_if_empty:N	1078
\tl_if_in:Nn	463, 749, 752
\tl_if_in:N	467
\tl_map_inline:nn	420, 990
\tl_new:Nn	688, 701, 1001–1017, 1019–1023, 1025–1028
\tl_put_right:cx	676
\tl_put_right:Nn	7, 696
\tl_put_right:N	1894
\tl_remove_all_in:Nn	699
\tl_remove_in:Nn	434, 1927
\tl_rescan:nn	1655, 1668
\tl_set:cn	30
\tl_set:cx	974
\tl_set:Nn	367–369, 371–373, 385–387, 389–391, 411, 413, 415, 505, 507–510, 549, 746–748, 762, 763, 766, 767, 976, 979, 982, 985, 1108–1112, 1772, 1774, 1805, 1807
\tl_set:No	1072–1074
\tl_set:Nv	1080
\tl_set:Nx	558, 1087, 1088, 1856, 1858
\tl_set_eq:NN	1857
\tl_use:c	1101
\to	1900
\token_if_eq_catcode_p:NN	1867, 1868
\token_if_eq_meaning_p:NN	1869

<code>\triangle</code>	1905	<code>\um_config_mathfrak_Latin:</code>	1287
<code>\trprime</code>	1707	<code>\um_config_mathfrak_latin:</code>	1295
<code>\typeout</code>	1061, 1066	<code>\um_config_mathit_Greek:</code>	1216
U			
<code>\um@backslash</code>	774, 797	<code>\um_config_mathit_greek:</code>	1226
<code>\um@char@num@range</code>	503, 682, 786, 787, 789	<code>\um_config_mathit_Latin:</code>	1194
<code>\um@firstchar</code>	773, 798	<code>\um_config_mathit_latin:</code>	1204
<code>\um@firstof</code>	796–798	<code>\um_config_mathit_misc:</code>	1236
<code>\um@parse@range</code>	783, 799	<code>\um_config_mathscr_Latin:</code>	1270
<code>\um@parse@term</code>	579, 615, 769, 838	<code>\um_config_mathscr_latin:</code>	1281
<code>\um@radicals</code>	701	<code>\um_config_mathsf_Latin:</code>	1326
<code>\um@scaled@apply</code>	439, 1746, 1752	<code>\um_config_mathsf_latin:</code>	1338
<code>\um@scanactivedef</code>	457, 1651	<code>\um_config_mathsfup_Latin:</code>	1302
<code>\um@scancharlet</code>	1651, 1680	<code>\um_config_mathsfup_latin:</code>	1314
<code>\um@zfb@feature</code>	704, 716, 719	<code>\um_config_mathsfup_num:</code>	1298
<code>\um_config_mathbb_Latin:</code>	1243	<code>\um_config_mathhtt_Latin:</code>	1353
<code>\um_config_mathbb_latin:</code>	1240	<code>\um_config_mathhtt_latin:</code>	1356
<code>\um_config_mathbb_misc:</code>	1256	<code>\um_config_mathhtt_num:</code>	1350
<code>\um_config_mathbb_num:</code>	1253	<code>\um_config_mathup_Greek:</code>	1170
<code>\um_config_mathbbit_misc:</code>	1263	<code>\um_config_mathup_greek:</code>	1180
<code>\um_config_mathbffrak_Latin:</code>	1509	<code>\um_config_mathup_Latin:</code>	1149
<code>\um_config_mathbffrak_latin:</code>	1512	<code>\um_config_mathup_latin:</code>	1159
<code>\um_config_mathbfit_Greek:</code>	1389	<code>\um_config_mathup_misc:</code>	1190
<code>\um_config_mathbfit_greek:</code>	1401	<code>\um_config_mathup_num:</code>	1145
<code>\um_config_mathbfit_Latin:</code>	1359	<code>\um_fontdimen_to_percent:nn</code>	436, 441, 444, 1748, 1749
<code>\um_config_mathbfit_latin:</code>	1374	<code>\um_glyph_if_exist:cT</code>	1090
<code>\um_config_mathbfit_misc:</code>	1415	<code>\um_glyph_if_exist:cTF</code>	1096
<code>\um_config_mathbfscr_Latin:</code>	1515	<code>\um_glyph_if_exist:n</code>	1123
<code>\um_config_mathbfscr_latin:</code>	1518	<code>\um_glyph_if_exist:nF</code>	1129
<code>\um_config_mathbfsfit_Greek:</code>	1612	<code>\um_glyph_if_exist:nT</code>	1128
<code>\um_config_mathbfsfit_greek:</code>	1624	<code>\um_glyph_if_exist:nTF</code>	1123, 1704, 1707, 1710
<code>\um_config_mathbfsfit_Latin:</code>	1588	<code>\um_glyph_if_exist_p:n</code>	1126
<code>\um_config_mathbfsfit_latin:</code>	1600	<code>\um_if_mathalph_decl:n</code>	745
<code>\um_config_mathbfsfit_misc:</code>	1636	<code>\um_if_mathalph_decl:nTF</code>	738
<code>\um_config_mathbfsfup_Greek:</code>	1549	<code>\um_init_alphabet:n</code>	554, 1079, 1118
<code>\um_config_mathbfsfup_greek:</code>	1561	<code>\um_make_mathactive:nNN</code>	625, 627
<code>\um_config_mathbfsfup_Latin:</code>	1525	<code>\um_map_char:cc</code>	864–869, 875, 886
<code>\um_config_mathbfsfup_latin:</code>	1537	<code>\um_map_char:nn</code>	881, 889
<code>\um_config_mathbfsfup_misc:</code>	1573	<code>\um_map_char:nn␣</code>	826
<code>\um_config_mathbfsfup_num:</code>	1521	<code>\um_map_char_internal:nn</code>	555, 563, 829
<code>\um_config_mathbfup_Greek:</code>	1464	<code>\um_map_char_noparse:nn</code>	555, 833, 839
<code>\um_config_mathbfup_greek:</code>	1476	<code>\um_map_char_parse:nn</code>	563, 837
<code>\um_config_mathbfup_Latin:</code>	1434	<code>\um_map_chars_Greek:nn</code>	872, 1172,
<code>\um_config_mathbfup_latin:</code>	1449		1175, 1218, 1221, 1392, 1396,
<code>\um_config_mathbfup_misc:</code>	1490		1467, 1471, 1551, 1555, 1614, 1618
<code>\um_config_mathbfup_num:</code>	1430		

\um_map_chars_greek:nn	. 861, 1182, 1185, 1228, 1231, 1404, 1408, 1479, 1483, 1563, 1567, 1626, 1630
\um_map_chars_Latin:nn 851, 1151, 1154, 1196, 1199, 1304, 1308, 1328, 1332, 1361, 1365, 1369, 1436, 1440, 1444, 1527, 1531, 1590, 1594
\um_map_chars_latin:nn 856, 1161, 1164, 1206, 1210, 1316, 1320, 1340, 1344, 1376, 1380, 1384, 1451, 1455, 1459, 1539, 1543, 1602, 1606
\um_map_chars_numbers:nn	.. 878, 1146
\um_map_chars_range:nnn 826, 843, 846, 849, 882
\um_map_chars_x:cc 879
\um_map_chars_x:nn 848, 892
\um_map_chars_xxiii:cc 863, 874
\um_map_chars_xxiii:nn 845, 890
\um_map_chars_xxiii:nn 826
\um_map_chars_xxvi:cc 853, 858
\um_map_chars_xxvi:nn 842, 891
\um_map_chars_xxvi:nn 826
\um_map_single:nnn	884, 1165, 1207, 1211
\um_mathmap:Nnn	552, 560, 896, 904, 1070
\um_mathmap_noparse:Nnn 552, 674, 684, 1070
\um_mathmap_parse:Nnn 560, 681
\um_maybe_init_alphabet:n 554, 562, 1054, 1079, 1091
\um_nprimes:n	1694, 1704, 1707, 1710, 1713
\um_nprimes_select:n 1700, 1731
\um_peek_execute_branches:ss: 1860, 1865
\um_peek_execute_branches_ss_aux: 1872, 1885
\um_prepare_alph:n 1120, 1130
\um_process_symbol_noparse:nnnn 551, 575
\um_process_symbol_parse:nnnn	559, 575
\um_remap_symbol:nnn 553, 561, 584, 585, 587, 590–593, 595, 596, 599–606, 608–611
\um_remap_symbol_noparse:nnn	553, 583
\um_remap_symbol_parse:nnn 561, 583, 614
\um_remap_symbols: 570, 583
\um_resolve_greek: 964
\um_scan_sscript:	1775, 1808, 1845, 1847
\um_scan_sscript:TF 1846, 1855
\um_scanprime: 1716, 1725, 1736, 1740, 1741
\um_scanprime_collect: 1718, 1720, 1723, 1726, 1729
\um_set_delcode:n	... 637, 640–666, 668
\um_set_delcode:nn	634–636, 638, 639, 668
\um_set_mathalph_range:Nnn 900
\um_set_mathalph_range:Nnn 900, 911, 914, 917
\um_set_mathalphabet_char:Ncc 923, 940, 946, 952–957
\um_set_mathalphabet_char:Nnn	893, 960
\um_set_mathalphabet_Greek:Nnn 943, 1178, 1224, 1390, 1393, 1397, 1465, 1468, 1472, 1552, 1556, 1559, 1615, 1619, 1622
\um_set_mathalphabet_greek:Nnn 949, 1188, 1234, 1402, 1405, 1411, 1477, 1480, 1486, 1564, 1568, 1571, 1627, 1631, 1634
\um_set_mathalphabet_Latin:Nnn 932, 1157, 1202, 1244, 1271, 1288, 1305, 1309, 1312, 1329, 1333, 1336, 1354, 1363, 1366, 1370, 1438, 1441, 1445, 1510, 1516, 1528, 1532, 1535, 1591, 1595, 1598
\um_set_mathalphabet_latin:Nnn 937, 1168, 1214, 1241, 1282, 1296, 1317, 1321, 1324, 1341, 1345, 1348, 1357, 1378, 1381, 1385, 1453, 1456, 1460, 1513, 1519, 1540, 1544, 1547, 1603, 1607, 1610
\um_set_mathalphabet_numbers:Nnn 927, 1147, 1254, 1299, 1300, 1351, 1431, 1432, 1522, 1523
\um_set_mathalphabet_pos:Nnnn 920, 1191, 1192, 1237, 1238, 1245–1251, 1257–1261, 1264–1268, 1272–1279, 1283–1285, 1289–1293, 1416, 1417, 1419,

1420, 1423, 1426, 1491–1496, 1498, 1499, 1502, 1505, 1574, 1575, 1577, 1578, 1581, 1584, 1637, 1638, 1640, 1641, 1644, 1647	\use:c 1097, 1115, 1133, 1141
\um_set_mathalphabet_x:Ncc 929	\use_i:nnn 1072
\um_set_mathalphabet_x:Nnn . 910, 963	\use_ii:nnn 1073
\um_set_mathalphabet_xxiii:Ncc 945, 951	\use_iii:nnn 1074
\um_set_mathalphabet_xxiii:Nnn 916, 961	\use_none:n 562
\um_set_mathalphabet_xxvi:Ncc 934, 939	\use_none:nnnn 1088
\um_set_mathalphabet_xxvi:Nnn 913, 962	\usv_set:nnn 29, 35–48, 50–63, 65–71, 73–79, 81–254
\um_set_mathcode:nnnn 496, 621, 677, 834	
\um_set_mathsymbol:nnNn 452, 576	V
\um_setup_active_subscript:nn 1799, 1813–1840	\varepsilon 982
\um_setup_active_superscript:nn 1766, 1780–1796	\varointclockwise 690
\um_setup_alphabets: 573, 1000	\varphi 985
\um_setup_delcodes: 572, 633	\vec 1901
\um_setup_math_alphabet:Nn 1086	\version@elt 424
\um_setup_math_alphabet:NV 1034–1050	\version@list 424
\um_setup_math_alphabet:VV 1082	\vysmwhtcircle 1907
\um_setup_math_mapping:n 1051–1053, 1055–1059, 1113	
\um_setup_mathactives: 571, 624	W
\um_setup_mathup: 1899	\widehat 1956
\um_setup_nabla: 365, 568	\widetilde 1957
\um_setup_partial: 383, 569	\wlog 1119
\um_split_arrow:w 750, 761	
\um_split_slash:w 753, 765	X
\um_sub_or_super:n .. 1773, 1806, 1849	\XeTeXdelcode ... 471, 477, 482, 669, 672
\um_symfont_tl 549, 558, 565, 576, 621, 629, 669, 672, 677, 835	\XeTeXdelimiter 470, 476, 483, 484
\um_tmp: 512, 515, 516, 535	\XeTeXmathaccent 487
\um_to_usv:nn 30, 32, 853	\XeTeXmathchardef 459, 628
\unicodecdots 1950	\XeTeXmathcharnum 1932, 1933
\unicodeellipsis 1949	\XeTeXmathcode .. 472, 478, 481, 489, 497
\UnicodeMathSymbol 551, 559, 1679	\XeTeXmathcodenum 631, 1769, 1802, 1932, 1933
\unimathsetup 255	\XeTeXradical 468
\unless 771	\XKV@rm 513, 531
\updefault 566	
\upDigamma 1960	Y
\updigamma 1959	\yen 1908
\upint 693	
	Z
	\Z 48, 1020
	\z 1019
	\z@ 1745, 1748, 1749, 1753
	\zf@family 566
	\zf@fontspec 517
	\zf@update@ff 717, 720