

Experimental unicode mathematical typesetting: The unicode-math package

Will Robertson

2009/09/30 v0.4

Abstract

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

Contents

1	Introduction	1	4.3	Other things	20
2	Unicode maths font setup	1	5	Fundamentals	21
2.1	Using multiple fonts	2	5.1	Enlarging the number of maths families	21
2.2	Script and scriptscript fonts/features	2	5.2	<code>\DeclareMathSymbol</code> for unicode ranges	21
3	Maths input	3	5.3	The main <code>\setmathfont</code> macro	23
3.1	Math ‘style’	3	5.4	(Big) operators	30
3.2	Bold style	3	5.5	Radicals	33
3.3	Sans serif style	5	5.6	Delimiters	34
3.4	All (the rest) of the mathematical alphabets	5	5.7	Maths accents	36
3.5	Miscellanea	6	6	Font features	37
I	The unicode-math package	11	6.1	OpenType maths font features	37
4	Things we need	11	6.2	Script and scriptscript font options	38
4.1	Package options	15	6.3	Range processing	38
4.2	Overcoming <code>\@on-lypreamble</code>	19	6.4	Resolving Greek symbol name control sequences	42

6.5	Setting up the mappings	44	II	stix table data extraction	67
7	Maths alphabets mapping definitions	45	A	Documenting maths support in the NFSS	69
7.1	Non-bold math alphabets	48	A.1	Overview	69
7.2	Bold math alphabets	51			
7.3	Definitions of the math symbols	58	III	X_YTeX math font dimensions	70
8	Epilogue	59			

1 Introduction

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

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

2 Unicode maths font setup

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

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

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

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

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

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

This has not been implemented yet.

Instances above of

```
[⟨font features⟩]{⟨font name⟩}
```

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

2.1 Using multiple fonts

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

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

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

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

2.2 Script and scriptscript fonts/features

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

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

3 Maths input

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

3.1 Math 'style'

Classically, \TeX uses italic lowercase Greek letters and *upright* uppercase Greek letters for variables in mathematics. This is contrary to the ISO standards of using italic forms for both upper- and lowercase. Furthermore, the French (contrary

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

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)	$(\alpha, \beta, \Gamma, \Xi)$

again, *quelle surprise*) have been known to use upright uppercase *Latin* letters as well as upright upper- and lowercase Greek.

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

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

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

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

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

3.2 Bold style

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

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

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

scarcity of bold italic fonts in the pre-digital typesetting era. It has been suggested that *italic* bold symbols are used nowadays instead.

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

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

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

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

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

The `bold-style` options' effects are shown in brief in table 2.

3.3 Sans serif style

Unicode contains upright and italic, medium and bold mathematical alphabet characters. These may be explicitly selected with the `\mathsfup`, `\mathsfit`, `\mathbf{sfup}`, and `\mathbf{sfit}` commands discussed in section §3.4.

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

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

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

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

3.3.1 What about bold sans serif?

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

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

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

3.4 All (the rest) of the mathematical alphabets

Unicode contains separate codepoints for most if not all variations of alphabet shape one may wish to use in mathematical notation. The complete list is shown in table 3. Some of these have been covered in the previous sections.

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

3.5 Miscellanea

3.5.1 Nabla

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

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

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

Font				Alphabet		
Style	Shape	Series	Switch	Latin	Greek	Numerals
Serif	Upright	Normal	<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>	•		•
Script	Upright	Normal	<code>\mathscr</code>	•		
		Bold	<code>\matbfscr</code>	•		
Fraktur	Upright	Normal	<code>\mathfrak</code>	•		
		Bold	<code>\mathbffrac</code>	•		

Table 4: The various forms of nabla.

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

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

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

3.5.2 Partial

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

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

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

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

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

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

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

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

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.

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.

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

3.5.4 Primes

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

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

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

3.5.5 Unicode subscripts and superscripts

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

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

3.5.6 Colon ‘:’

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

$\text{A}_0 \text{ }_1 \text{ }_2 \text{ }_3 \text{ }_4 \text{ }_5 \text{ }_6 \text{ }_7 \text{ }_8 \text{ }_9 \text{ }_{+} \text{ }_{-} \text{ }_{=} \text{ }_{(} \text{ }_{)} \text{ }_a \text{ }_e \text{ }_i \text{ }_o \text{ }_r \text{ }_u \text{ }_v \text{ }_x \text{ }_\beta \text{ }_\gamma \text{ }_\rho \text{ }_\varphi \text{ }_\chi \text{ }_Z$

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

Table 6: Slashes and backslashes.

Slot	Name	Glyph	Command
U+002F	SOLIDUS	/	<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>

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

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

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

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

3.5.7 Slashes and backslashes

There are several slash-like symbols defined in unicode. These are shown in table 6. The ASCII slashes / and \ are useful as input characters but should not be used in the rendering of mathematics. (I think.)

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.

If encountered in the input stream, therefore, I believe it should be mapped to the meaning of U+2215: DIVISION SLASH. (Alas, see the note below.)

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

I do not know what U+29F8: BIG SOLIDUS is intended to be used for. It's a 'math operator' (like Σ) so it falls outside the topic of discussion here.

Backslash MathML uses U+2216: SET MINUS like this: $A \setminus B$.² I think the STIX name for this glyph slot should just be `\setminus`.

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': $\pi \approx 7 \backslash 22$.³

Again, I don't know what U+29F9: BIG REVERSE SOLIDUS is for. But it's not too important at this stage.

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

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

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

And none of the backslashes stretch. Which leaves me in a bit of a pickle. \TeX has a stretchy backslash. Cambria Math does not. What will? And in which glyph slot? I give up, for now. This is an impossible problem.

All of the above characters are allowed to be used after `\left`, `\middle`, and `\right`. Only the font will know whether or not it will actually stretch, however. If you like you may redefine `\slash` and `\backslash` to fit your needs. Perhaps this will be a package option some day.

3.5.8 Normalising some input characters

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

²§4.4.5.11 [Unicode: // 2003. 03. 000/00/0000003/](#)

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

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

(Not yet implemented.)

File I

The unicode-math package

This is the package.

```

1 \ProvidesPackage{unicode-math}
2 [2009/09/30 v0.4 Unicode maths in XeLaTeX]

```

4 Things we need

Packages

```

3 \RequirePackage{expl3}[2009/08/12]
4 \RequirePackage{xparse}[2009/08/31]
5 \RequirePackage{fontspec}
   Start using LATEX3 — finally!
6 \ExplSyntaxOn

```

Counters and conditionals

```

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

```

For math-style:

```

10 \newif\if@um@literal
11 \newif\if@um@upGreek
12 \newif\if@um@upgreek

```

```

13 \newif\if@um@upLatin
14 \newif\if@um@uplatin

```

For bold-style:

```

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

```

For nabla:

```

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

```

4.0.9 Alphabet unicode positions

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

```

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

```

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

```

49 \def\um@usv@ttLatin{"1D670}
50 \def\um@usv@ttlatin{"1D68A}

```

Bold:

```

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

```

Greek variants:

```

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

```

93 \def\um@usv@digamma{"3DD}

Bold:

94 \def\um@usv@bfvarTheta{"1D6B9}

95 \def\um@usv@bfDigamma{"1D7CA}

96 \def\um@usv@bfvarepsilon{"1D6DC}

97 \def\um@usv@bfvartheta{"1D6DD}

98 \def\um@usv@bfvarkappa{"1D6DE}

99 \def\um@usv@bfvarphi{"1D6DF}

100 \def\um@usv@bfvarrho{"1D6E0}

101 \def\um@usv@bfvarpi{"1D6E1}

102 \def\um@usv@bfdigamma{"1D7CB}

Italic Greek variants:

103 \def\um@usv@ith{"210E}

104 \def\um@usv@itvarTheta{"1D6F3}

105 \def\um@usv@itvarepsilon{"1D716}

106 \def\um@usv@itvartheta{"1D717}

107 \def\um@usv@itvarkappa{"1D718}

108 \def\um@usv@itvarphi{"1D719}

109 \def\um@usv@itvarrho{"1D71A}

110 \def\um@usv@itvarpi{"1D71B}

Bold italic:

111 \def\um@usv@bfuph{"1D421}

112 \def\um@usv@bfith{"1D489}

113 \def\um@usv@bfitvarTheta{"1D72D}

114 \def\um@usv@bfitvarepsilon{"1D750}

115 \def\um@usv@bfitvartheta{"1D751}

116 \def\um@usv@bfitvarkappa{"1D752}

117 \def\um@usv@bfitvarphi{"1D753}

118 \def\um@usv@bfitvarrho{"1D754}

119 \def\um@usv@bfitvarpi{"1D755}

Nabla:

120 \def\um@usv@Nabla{"2207}

121 \def\um@usv@itNabla{"1D6FB}

122 \def\um@usv@bfNabla{"1D6C1}

123 \def\um@usv@bfitNabla{"1D735}

124 \def\um@usv@bfsfNabla{"1D76F}

125 \def\um@usv@bfsfitNabla{"1D7A9}

Partial:

126 \def\um@usv@partial{"2202}

127 \def\um@usv@itpartial{"1D715}

128 \def\um@usv@bfpartial{"1D6DB}

129 \def\um@usv@bfitpartial{"1D74F}

130 \def\um@usv@bfsfpartial{"1D789}

131 \def\um@usv@bfsfitpartial{"1D7C3}

4.1 Package options

xkeyval's package support is used here.

math-style

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



```

173 \or
174 \um@literaltrue
175 \um@bfliteraltrue
176 \bool_set_true:N \g_um_sfliteral_bool
177 \bool_set_false:N \g_um_texgreek_bool
178 \fi
179 }

```

bold-style

```

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

```

sans-style

```

209 \bool_new:N \g_um_upsans_bool
210 \bool_new:N \g_um_sfliteral_bool
211 \define@choicekey*{unicode-math.sty}
212 {sans-style}[\@tempa\@tempb]{iso,tex,literal}{
213 \ifcase\@tempb\relax

```

```

214 \bool_set_false:N \g_um_upsans_bool
215 \or
216 \bool_set_true:N \g_um_upsans_bool
217 \or
218 \bool_set_true:N \g_um_sfliteral_bool
219 \fi
220 }
221 \cs_set:Nn \um_setup_sfshapes: {
222 \bool_if:NTF \g_um_upsans_bool {
223 \tl_set:Nn \um_sf_Latin_up_or_it_usv { \um@usv@sflatin }
224 \tl_set:Nn \um_sf_latin_up_or_it_usv { \um@usv@sflatin }
225 \tl_set:Nn \um_bfsf_Latin_up_or_it_usv { \um@usv@bfsfupLatin }
226 \tl_set:Nn \um_bfsf_latin_up_or_it_usv { \um@usv@bfsfuplatin }
227 \tl_set:Nn \um_bfsf_Greek_up_or_it_usv { \um@usv@bfsfupGreek }
228 \tl_set:Nn \um_bfsf_greek_up_or_it_usv { \um@usv@bfsfupgreek }
229 }{
230 \tl_set:Nn \um_sf_Latin_up_or_it_usv { \um@usv@sfitLatin }
231 \tl_set:Nn \um_sf_latin_up_or_it_usv { \um@usv@sfitlatin }
232 \tl_set:Nn \um_bfsf_Latin_up_or_it_usv { \um@usv@bfsfitLatin }
233 \tl_set:Nn \um_bfsf_latin_up_or_it_usv { \um@usv@bfsfitlatin }
234 \tl_set:Nn \um_bfsf_Greek_up_or_it_usv { \um@usv@bfsfitGreek }
235 \tl_set:Nn \um_bfsf_greek_up_or_it_usv { \um@usv@bfsfitgreek }
236 }
237 }

```

Symbol obliqueness

```

238 \define@choicekey*{unicode-math.sty}{nabla}[\@tempa\@tempb]{upright,italic}{
239 \ifcase\@tempb\relax
240 \um@upNablatrue
241 \or
242 \um@upNablafalse
243 \fi
244 }
245 \cs_set:Nn \um_setup_nabla: {
246 \if@um@upNabla
247 \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@Nabla }
248 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfNabla }
249 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfNabla }
250 \else
251 \tl_set:Nn \um_Nabla_up_or_it_usv { \um@usv@itNabla }
252 \tl_set:Nn \um_bfNabla_up_or_it_usv { \um@usv@bfitNabla }
253 \tl_set:Nn \um_bfsfNabla_up_or_it_usv { \um@usv@bfsfitNabla }
254 \fi
255 }
256 \define@choicekey*{unicode-math.sty}{partial}[\@tempa\@tempb]{upright,italic}{
257 \ifcase\@tempb\relax

```

```

258     \@um@uppartialtrue
259   \or
260     \@um@uppartialfalse
261   \fi
262 }
263 \cs_set:Nn \um_setup_partial: {
264   \if@um@uppartial
265     \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@partial }
266     \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfpartial }
267     \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfpartial }
268   \else
269     \tl_set:Nn \um_partial_up_or_it_usv { \um@usv@itpartial }
270     \tl_set:Nn \um_bfpartial_up_or_it_usv { \um@usv@bfitpartial }
271     \tl_set:Nn \um_bfsfpartial_up_or_it_usv { \um@usv@bfsfitpartial }
272   \fi
273 }

```

Epsilon and phi shapes

```

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

```

Colon style

```

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

```

4.2 Overcoming \@onlypreamble

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

```

291 \tl_map_inline:nn {
292   \new@mathgroup

```

```

293 \cdp@list
294 \cdp@elt
295 \DeclareMathSizes
296 \@DeclareMathSizes
297 \newmathalphabet
298 \newmathalphabet@@
299 \newmathalphabet@@@
300 \DeclareMathVersion
301 \define@mathalphabet
302 \define@mathgroup
303 \addtoversion
304 \version@list
305 \version@elt
306 \alpha@list
307 \alpha@elt
308 \restore@mathversion
309 \init@restore@version
310 \dorestore@version
311 \process@table
312 \new@mathversion
313 \DeclareSymbolFont
314 \group@list
315 \group@elt
316 \new@symbolfont
317 \SetSymbolFont
318 \SetSymbolFont@
319 \get@cdp
320 \DeclareMathAlphabet
321 \new@mathalphabet
322 \SetMathAlphabet
323 \SetMathAlphabet@
324 \DeclareMathAccent
325 \set@mathaccent
326 \DeclareMathSymbol
327 \set@mathchar
328 \set@mathsymbol
329 \DeclareMathDelimiter
330 \@xxDeclareMathDelimiter
331 \@DeclareMathDelimiter
332 \@xDeclareMathDelimiter
333 \set@mathdelimiter
334 \set@@mathdelimiter
335 \DeclareMathRadical
336 \mathchar@type
337 \DeclareSymbolFontAlphabet
338 \DeclareSymbolFontAlphabet@

```

```

339 }{
340 \tl_remove_in:Nn \@preamblecmds {\do#1}
341 }

```

4.3 Other things

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

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

0.73
0.60
0.65

```

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

```

```

342 \def\um@fontdimen@percent#1#2{
343   0.\strip@pt\dimexpr\fontdimen#1#2 *65536\relax
344 }

```

`\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.

```

345 \def\um@scaled@apply#1#2#3{
346   \ifx#1\scriptstyle
347     #2\um@fontdimen@percent{10}\um@font#3
348   \else
349     \ifx#1\scriptscriptstyle
350       #2\um@fontdimen@percent{11}\um@font#3
351     \else
352       #2#3%
353     \fi
354   \fi
355 }

```

5 Fundamentals

5.1 Enlarging the number of maths families

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

```

356 \def\new@mathgroup{\alloc@8\mathgroup\chardef\@cclvi}
357 \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`.

5.2 `\DeclareMathSymbol` for unicode ranges

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

```
\um@mathsymbol #1 : Symbol, e.g., \alpha
                #2 : Type, e.g., \mathalpha
                #3 : Math font name, e.g., operators
                #4 : Slot, e.g., "221E
358 \def \um@mathsymbol#1#2#3#4{
359   \expandafter\um@set@mathsymbol\csname sym#3\endcsname#1#2{#4}}
```

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

```
\um@set@mathsymbol #1 : Symbol font number
                   #2 : Symbol macro, e.g., \alpha
                   #3 : Type, e.g., \mathalpha
                   #4 : Slot, e.g., "221E
If the symbol definition is for a macro. There are a bunch of tests to perform to
process the various characters.
360 \def\um@set@mathsymbol#1#2#3#4{
```

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

```
361 \ifx\mathop#3\relax
```

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

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

```
362 \begingroup
363   \char_make_active:n {#4}
364   \global\mathcode#4="8000\relax
365   \um@scanactivedef #4 \@nil { \csname\cs_to_str:N #2 op\endcsname }
366 \endgroup
```

Some of these require a `\nolimits` suffix. This is controlled by the `\um@nolimits` macro, which contains a list of such characters. This list is checked dynamically because we're not interested in efficiency. Or something. This allows the list to be updated in the middle of a document.

Declare the plain old mathchardef for the control sequence `\sum@sym`.

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

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

```

370 \cs_gset:cpn { \cs_to_str:N #2 op } {
371   \csname\string#2@sym\endcsname
372   \expandafter\in@\expandafter#2\expandafter{\um@nolimits}
373   \ifin@
374   \expandafter\nolimits
375   \fi
376 }

```

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

```

377 \else

```

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

```

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

```

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

```

382 \ifx\mathopen#3\relax
383   \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
384   \global\XeTeXdelcode#4=#1 #4\relax
385   \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
386 \else
387   \ifx\mathclose#3\relax
388     \cs_gset:Npn #2 {\XeTeXdelimiter "\mathchar@type#3 #1 #4\relax}
389     \global\XeTeXdelcode#4=#1 #4\relax
390     \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
391   \else

```

Accents

```

392 \ifx\mathaccent#3\relax
393   \cs_gset:Npx #2 {\XeTeXmathaccent "\mathchar@type#3 #1 #4\relax}
394 \else

```

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

```

395   \global\XeTeXmathcode#4="\mathchar@type#3 #1 #4\relax
396   \fi
397 \fi
398 \fi
399 \fi

```

```

400 \fi
401 }

```

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

```

402 \cs_set:Nn \um_set_mathcode:nnnn {
403   \XeTeXmathcode#1="\mathchar@type#2 \csname sym#3\endcsname #4\relax
404 }

```

5.3 The main `\setmathfont` macro

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

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

```

405 \DeclareDocumentCommand \setmathfont { O{ } m } {

```

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

```

406   \let\glb@currsizel\relax

```

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

```

407   \let\um@char@range\@empty
408   \let\um@char@num@range\@empty

```

- Tell fontspec that maths font features are actually allowed.

```

409   \@um@fontspec@featuretrue

```

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

```

410   \csname S@\f@size\endcsname

```

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

```

411   \def\um@mversion{normal}
412   \DeclareMathVersion{\um@mversion}

```


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

```

413 \tl_set:Nn \l_um_script_features_tl {ScriptStyle}
414 \tl_set:Nn \l_um_sscript_features_tl {ScriptScriptStyle}
415 \tl_set:Nn \l_um_script_font_tl      {#2}
416 \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.

```

417 \setkeys*[um]{options}{#1}
418 \edef\@tempa{\noexpand\zf@fontspec{
419     Script = Math,
420     SizeFeatures = {
421         {Size = \tf@size-} ,
422         {Size = \sf@size-\tf@size ,
423           Font = \l_um_script_font_tl ,
424             \l_um_script_features_tl
425         } ,
426         {Size = -\sf@size ,
427           Font = \l_um_sscript_font_tl ,
428             \l_um_sscript_features_tl
429         }
430     },
431     \XKV@rm
432 }{#2}
433 }
434 \@tempa

```

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

Check for the correct number of `\fontdimens`:

```

435 \font\um@font="#2"\relax
436 %% \ifdim \dimexpr\fontdimen9\um@font*65536\relax =65pt\relax
437 %% \@um@ot@math@true
438 %% \else
439 %% \PackageWarningNoLine{unicode-math}{
440 %%     The~ font~ '#2' ~is~ not~ a~ valid~ OpenType~ maths~ font.~
441 %%     Some~ maths~ features~ will~ not~ be~ available~ or~ behave~
442 %%     in~ a~ substandard~ manner
443 %% }
444 %% \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 §5.3.1 for the individual definitions

```

445 \ifx\um@char@range\@empty
446   \tl_set:Nn \um_symfont_tl {um@allsym}
447   \PackageInfo{unicode-math}{Defining~ the~ default~ maths~ font~ as~ '#2'}
448   \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_noparse:nnnn
449   \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_noparse:Nnn
450   \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_noparse:nnn
451   \cs_set_eq:NN \um_maybe_init_alphabet:n \um_init_alphabet:n
452 \else
453   \stepcounter{um@fam}
454   \tl_set:Nx \um_symfont_tl {um@fam\theum@fam}
455   \cs_set_eq:NN \UnicodeMathSymbol \um_process_symbol_parse:nnnn
456   \cs_set_eq:NN \um_mathmap:Nnn \um_mathmap_parse:Nnn
457   \cs_set_eq:NN \um_remap_symbol:nnn \um_remap_symbol_parse:nnn
458   \cs_set_eq:NN \um_maybe_init_alphabet:n \use_none:n
459 \fi

```

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

```

460 \DeclareSymbolFont{\um_symfont_tl}
461   {\encodingdefault}{\zf@family}{\mddefault}{\updefault}

```

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

```

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

```

Finally,

- Set up shapes for italic/upright or ordinary/var symbols as per package options.
- Remap symbols that don't take their natural mathcode
- Activate any symbols that need to be math-active
- Setup all symbols not covered by the table (mostly alphanumerics)
- Setup the maths alphabets (`\mathbf` etc.)

```

463 \um_setup_shapes:
464 \um_remap_symbols:
465 \um_setup_mathactives:
466 \um_setup_alphanum:
467 \um_setup_alphabets:

```

End of the `\setmathfont` macro.

```

468 }

469 \cs_new:Nn \um_setup_shapes: {
470   \um_setup_nabla:
471   \um_setup_partial:
472   \um_setup_sfshapes:
473   \um_setup_bfshapes:
474 }

```

5.3.1 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 §6.3 for the code that enables this.

```

475 \cs_set:Nn \um_process_symbol_noparse:nnnn {
476   \um@mathsymbol{#2}{#3}{\um_symfont_t1}{#1}
477 }

478 \cs_set:Nn \um_process_symbol_parse:nnnn {
479   \um@parse@term{#1}{#2}{#3}{
480     \um_process_symbol_noparse:nnnn{#1}{#2}{#3}{#4}
481   }
482 }
```

`\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:nnnn`

```

483 \cs_new:Nn \um_remap_symbols: {
484   \um_remap_symbol:nnn{`-}{\mathbin}{"02212}% hyphen to minus
485   \um_remap_symbol:nnn{`*}{\mathbin}{"02217}% text asterisk to "cen-
      tred asterisk"
486   \bool_if:NF \g_um_literal_colon_bool {
487     \um_remap_symbol:nnn{`\:}{\mathrel}{"02236}% colon to ratio (i.e., punct to rel)
488   }
489   \if@um@literal
490     \um_remap_symbol:nnn {\um@usv@Nabla}{\mathord}{\um@usv@Nabla}
491     \um_remap_symbol:nnn {\um@usv@itNabla}{\mathord}{\um@usv@itNabla}
492     \um_remap_symbol:nnn {\um@usv@partial}{\mathord}{\um@usv@partial}
493     \um_remap_symbol:nnn {\um@usv@itpartial}{\mathord}{\um@usv@itpartial}
494   \else
495     \um_remap_symbol:nnn {\um@usv@Nabla,\um@usv@itNabla}{\mathord}{\um_Nabla_up_or_it_usv}
496     \um_remap_symbol:nnn {\um@usv@partial,\um@usv@itpartial}{\mathord}{\um_partial_up_or_it_usv}
497   \fi
```

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

```

498 \if@um@bfliteral
499   \um_remap_symbol:nnn {\um@usv@bfNabla}{\mathord}{\um@usv@bfNabla}
500   \um_remap_symbol:nnn {\um@usv@bfitNabla}{\mathord}{\um@usv@bfitNabla}
501   \um_remap_symbol:nnn {\um@usv@bfsfNabla}{\mathord}{\um@usv@bfsfNabla}
502   \um_remap_symbol:nnn {\um@usv@bfsfitNabla}{\mathord}{\um@usv@bfsfitNabla}
503   \um_remap_symbol:nnn {\um@usv@bfpartial}{\mathord}{\um@usv@bfpartial}
504   \um_remap_symbol:nnn {\um@usv@bfitpartial}{\mathord}{\um@usv@bfitpartial}
505   \um_remap_symbol:nnn {\um@usv@bfsfpartial}{\mathord}{\um@usv@bfsfpartial}
506   \um_remap_symbol:nnn {\um@usv@bfsfitpartial}{\mathord}{\um@usv@bfsfitpartial}
507 \else
508   \um_remap_symbol:nnn {\um@usv@bfNabla,\um@usv@bfitNabla}{\mathord}{\um_bfNabla_up_or_it_usv}
509   \um_remap_symbol:nnn {\um@usv@bfsfNabla,\um@usv@bfsfitNabla}{\mathord}{\um_bfsfNabla_up_or_it_usv}
510   \um_remap_symbol:nnn {\um@usv@bfpartial,\um@usv@bfitpartial}{\mathord}{\um_bfpartial_up_or_it_usv}
```

```

511 \um_remap_symbol:nnn {\um@usv@bfsfpartial,\um@usv@bfsfitpartial}{\mathord}{\um_bfsfpartia
512 \fi
513 }

```

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

```

514 \cs_new:Nn \um_remap_symbol_parse:nnn {
515   \um@parse@term {#3} {\@nil} {#2} {
516     \um_remap_symbol_noparse:nnn {#1} {#2} {#3}
517   }
518 }
519 \cs_new:Nn \um_remap_symbol_noparse:nnn {
520   \clist_map_inline:nn {#1} {
521     \um_set_mathcode:nnnn {##1} {#2} {\um_symfont_tl} {#3}
522   }
523 }

```

5.3.2 Active math characters

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

`\um_setup_mathactives:`

```

524 \cs_new:Nn \um_setup_mathactives: {
525   \um_make_mathactive:nnn {"2032} \primesingle \mathord
526 }

```

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

```

527 \cs_new:Nn \um_make_mathactive:nnn {
528   \XeTeXmathchardef #2 = "\mathchar@type #3
529                               \csname sym\um_symfont_tl\endcsname
530                               #1 \scan_stop:
531   \XeTeXmathcodenum #1 = "1FFFFF \scan_stop:
532 }

```

5.3.3 Maths alphabets' character mapping

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

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

```

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

```

Normal weight

```
536 \if@um@literal
537 \um_setup_literals:
538 \else
539 \um_setup_Latin:
540 \um_setup_latin:
541 \um_setup_Greek:
542 \um_setup_greek:
543 \fi
```

Bold

```
544 \if@um@bfliteral
545 \um_setup_bf_literals:
546 \else
547 \if@um@bfupLatin
548 \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfupLatin}
549 \else
550 \um_map_chars_latin:nn {\um@usv@bfupLatin,\um@usv@bfitLatin}{\um@usv@bfitLatin}
551 \fi
552 \if@um@bfuplatin
553 \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfuplatin}
554 \else
555 \um_map_chars_latin:nn {\um@usv@bfuplatin,\um@usv@bfitlatin}{\um@usv@bfitlatin}
556 \fi
557 \if@um@bfupGreek
558 \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfupGreek}
559 \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfvarTheta}
560 \else
561 \um_map_chars_greek:nn {\um@usv@bfupGreek,\um@usv@bfitGreek}{\um@usv@bfitGreek}
562 \um_map_char:nn {\um@usv@bfvarTheta,\um@usv@bfitvarTheta}{\um@usv@bfitvarTheta}
563 \fi
564 \if@um@bfupgreek
565 \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfupgreek}
566 \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfvarepsilon}
567 \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfvartheta}
568 \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfvarkappa}
569 \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfvarphi}
570 \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfvarrho}
571 \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfvarpi}
572 \else
573 \um_map_chars_greek:nn {\um@usv@bfupgreek,\um@usv@bfitgreek}{\um@usv@bfitgreek}
574 \um_map_char:nn {\um@usv@bfvarepsilon,\um@usv@bfitvarepsilon}{\um@usv@bfitvarepsilon}
575 \um_map_char:nn {\um@usv@bfvartheta,\um@usv@bfitvartheta}{\um@usv@bfitvartheta}
576 \um_map_char:nn {\um@usv@bfvarkappa,\um@usv@bfitvarkappa}{\um@usv@bfitvarkappa}
577 \um_map_char:nn {\um@usv@bfvarphi,\um@usv@bfitvarphi}{\um@usv@bfitvarphi}
578 \um_map_char:nn {\um@usv@bfvarrho,\um@usv@bfitvarrho}{\um@usv@bfitvarrho}
```

```

579     \um_map_char:nn {\um@usv@bfvarpi,\um@usv@bfitvarpi}{\um@usv@bfitvarpi}
580     \fi
581   \fi
582   \else
: TODO : what is supposed to happen here?
583   \fi
584 }

```

5.3.4 Functions for setting up the maths alphabets

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

```

585 \cs_set:Nn \um_mathmap_noparse:Nnn {
586   \clist_map_inline:nn {#2} {
587     \exp_args:No \um@addto@mathmap \um_symfont_tl {##1}{#1}{#3}
588   }
589 }

```

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

```

590 \cs_set:Nn \um_mathmap_parse:Nnn {
591   \clist_map_inline:Nn \um@char@num@range {
592     \ifnum##1=#3\relax
593       \clist_map_inline:nn {#2} {
594         \exp_args:No \um@addto@mathmap \um_symfont_tl {####1}{#1}{#3}
595       }
596     \fi
597   }
598 }

```

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

This macro is used so that `\um_symfont_t1` can be expanded before entering the `\g@addto@macro` command.

```

599 \newcommand\um@addto@mathmap[4]{
600   \tl_put_right:cn {um_setup\_cs_to_str:N #3:} {
601     \um_set_mathcode:nnnn{#2}{\mathalpha}{#1}{#4}
602   }
603 }
```

5.4 (Big) operators























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

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

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

usv	Ex.	Macro	Description
U+02140	$\sum\limits_{0,1}$	<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>\ointctrackwise</code>	CONTOUR INTEGRAL, ANTICLOCKWISE
U+022C0	$\bigwedge\limits_{0,1}$	<code>\bigwedge</code>	LOGICAL OR OPERATOR

U+022C1	\bigvee	<code>\bigvee</code>	LOGICAL AND OPERATOR
U+022C2	\bigcap	<code>\bigcap</code>	INTERSECTION OPERATOR
U+022C3	\bigcup	<code>\bigcup</code>	UNION OPERATOR
U+027D5	\leftthreetimes	<code>\leftthreetimes</code>	LEFT OUTER JOIN
U+027D6	\rthreetimes	<code>\rthreetimes</code>	RIGHT OUTER JOIN
U+027D7	\fullthreetimes	<code>\fullthreetimes</code>	FULL OUTER JOIN
U+027D8	\Uparrow	<code>\biguparrow</code>	LARGE UP TACK
U+027D9	\Downarrow	<code>\bigdownarrow</code>	LARGE DOWN TACK
U+029F8	$\big/$	<code>\xsol</code>	BIG SOLIDUS
U+029F9	$\big\backslash$	<code>\xbsol</code>	BIG REVERSE SOLIDUS
U+02A00	\bigodot	<code>\bigodot</code>	N-ARY CIRCLED DOT OPERATOR
U+02A01	\bigoplus	<code>\bigoplus</code>	N-ARY CIRCLED PLUS OPERATOR
U+02A02	\bigotimes	<code>\bigotimes</code>	N-ARY CIRCLED TIMES OPERATOR
U+02A03	$\bigcup\!\!\!\cdot$	<code>\bigcupdot</code>	N-ARY UNION OPERATOR WITH DOT
U+02A04	$\bigcup\!\!\!+$	<code>\bigcupplus</code>	N-ARY UNION OPERATOR WITH PLUS
U+02A05	\bigsqcap	<code>\bigsqcap</code>	N-ARY SQUARE INTERSECTION OPERATOR
U+02A06	\bigsqcup	<code>\bigsqcup</code>	N-ARY SQUARE UNION OPERATOR
U+02A07	\And	<code>\conjquant</code>	TWO LOGICAL AND OPERATOR
U+02A08	\Or	<code>\disjquant</code>	TWO LOGICAL OR OPERATOR
U+02A09	\bigtimes	<code>\bigtimes</code>	N-ARY TIMES OPERATOR
U+02A0B	\sumint	<code>\sumint</code>	SUMMATION WITH INTEGRAL
U+02A0C	\iiint	<code>\iiiint</code>	QUADRUPLE INTEGRAL OPERATOR
U+02A0D	\int	<code>\intbar</code>	FINITE PART INTEGRAL

U+02A0E		<code>\intBar</code>	INTEGRAL WITH DOUBLE STROKE
U+02A0F		<code>\fint</code>	INTEGRAL AVERAGE WITH SLASH
U+02A10		<code>\cirfnint</code>	CIRCULATION FUNCTION
U+02A11		<code>\awint</code>	ANTICLOCKWISE INTEGRATION
U+02A12		<code>\rppolint</code>	PATH AROUND POLE LINE INTEGRATION WITH SEMICIRCULAR
U+02A13		<code>\scpolint</code>	PATH AROUND POLE LINE INTEGRATION NOT INCLUDING THE
U+02A14		<code>\npolint</code>	POLE
U+02A15		<code>\pointint</code>	INTEGRAL AROUND A POINT OPERATOR
U+02A16		<code>\sqint</code>	QUATERNION INTEGRAL OPERATOR INTEGRAL WITH LEFTWARDS ARROW WITH
U+02A17		<code>\intlarhk</code>	HOOK
U+02A18		<code>\intx</code>	INTEGRAL WITH TIMES SIGN
U+02A19		<code>\intcap</code>	INTEGRAL WITH INTERSECTION
U+02A1A		<code>\intcup</code>	INTEGRAL WITH UNION
U+02A1B		<code>\upint</code>	INTEGRAL WITH OVERBAR
U+02A1C		<code>\lowint</code>	INTEGRAL WITH UNDERBAR
U+02A1D		<code>\Join</code>	JOIN
U+02A1E		<code>\bigtriangleleft</code>	LARGE LEFT TRIANGLE OPERATOR
U+02A1F		<code>\zcmp</code>	Z NOTATION SCHEMA COMPOSITION
U+02A20		<code>\zpipe</code>	Z NOTATION SCHEMA PIPING
U+02A21		<code>\zproject</code>	Z NOTATION SCHEMA PROJECTION
U+02AFC		<code>\biginterleave</code>	LARGE TRIPLE VERTICAL BAR OPERATOR
U+02AFF		<code>\bigtalloblong</code>	N-ARY WHITE VERTICAL BAR

`\um@nolimits` This macro is a sequence containing those maths operators that require a `\nolimits` suffix. This list is used when processing `unicode-math-table.tex` to define such commands automatically (see the macro `\um@set@mathsymbol` on page 21). 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 \iiint , but that might be a matter of preference.

```

604 \def\um@nolimits{
605   \@elt\int\@elt\iint\@elt\iiint\@elt\iiint\@elt\oint\@elt\oint\@elt\oiint
606   \@elt\intclockwise\@elt\varointclockwise\@elt\ointctrclockwise\@elt\sumint
607   \@elt\intbar\@elt\intBar\@elt\fint\@elt\cirfnint\@elt\awint\@elt\rppolint

```

```

608 \@elt\scpolint\@elt\npolint\@elt\pointint\@elt\sqint\@elt\intlarkh\@elt\intx
609 \@elt\intcap\@elt\intcup\@elt\upint\@elt\lowint
610 }

```

\addnolimits This macro appends material to the macro containing the list of operators that don't take limits. See example following for usage. Note at present that this command must have taken effect before `\setmathfont`.

```

611 \newcommand\addnolimits[1]{
612   \expandafter\def\expandafter\um@nolimits\expandafter{\um@nolimits\@elt#1}
613 }

```

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

```

614 \def\removenolimits#1{
615   \begingroup
616     \def\@elt##1{
617       \ifx##1#1\else
618         \noexpand\@elt\noexpand##1
619       \fi}
620     \xdef\um@nolimits{\um@nolimits}
621   \endgroup
622 }

```

5.5 Radicals

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

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

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

```

623 \def\um@radicals{\sqrt}

```

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

```

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

```

5.6 Delimiters

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

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

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

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












Some symbols that aren't `mathopen`/`mathclose` still need to have delimiter codes assigned.

Set up `delcodes` so that slashes and things can grow if the font supports it. This is just inserted here so the documentation works. It will be generalised soon.

```
626 \XeTeXdelcode"002F =4 "002F % ord
627 \XeTeXdelcode"005C =4 "005C % ord
628 \XeTeXdelcode"2044 =4 "2044 % bin
629 \XeTeXdelcode"2215 =4 "2215 % bin
630 \XeTeXdelcode"2216 =4 "2216 % bin
631 \XeTeXdelcode"29F5 =4 "29F5 % bin
```








Here are all `\mathopen` characters:

USV	Ex.	Macro	Description
U+00028	(<code>\lparen</code>	LEFT PARENTHESIS
U+0005B	[<code>\lbrack</code>	LEFT SQUARE BRACKET
U+0007B	{	<code>\lbrace</code>	LEFT CURLY BRACKET
U+0007C		<code>\lvert</code>	VERTICAL BAR
U+02016		<code>\lVert</code>	DOUBLE VERTICAL BAR
U+0221A	√	<code>\sqrt</code>	RADICAL
U+0221B	∛	<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
U+0298D		<code>\lbrackultick</code>	CORNER LEFT SQUARE BRACKET WITH TICK IN
U+0298F		<code>\lbracklltick</code>	BOTTOM CORNER
U+02991		<code>\langedot</code>	LEFT ANGLE BRACKET WITH DOT
U+02993		<code>\lparenless</code>	LEFT ARC LESS-THAN BRACKET
U+02997		<code>\lblrbrak</code>	LEFT BLACK TORTOISE SHELL BRACKET
U+029D8		<code>\lvzigzag</code>	LEFT WIGGLY FENCE
U+029DA		<code>\lvzigzag</code>	LEFT DOUBLE WIGGLY FENCE
U+029FC		<code>\lcurvyangle</code>	LEFT POINTING CURVED ANGLE BRACKET
U+03014		<code>\lbrbrak</code>	LEFT BROKEN BRACKET
U+03018		<code>\Lbrbrak</code>	LEFT WHITE TORTOISE SHELL BRACKET

























And `\mathclose`:

USV	Ex.	Macro	Description
U+00029)	<code>\rparen</code>	RIGHT PARENTHESIS
U+0005D]	<code>\rbrack</code>	RIGHT SQUARE BRACKET
U+0007C		<code>\rvert</code>	VERTICAL BAR
U+0007D	}	<code>\rbrace</code>	RIGHT CURLY BRACKET
U+02016		<code>\rVert</code>	DOUBLE VERTICAL BAR
U+02309]̂	<code>\rceil</code>	RIGHT CEILING
U+0230B]̇	<code>\rfloor</code>	RIGHT FLOOR
U+0231D	┐	<code>\urcorner</code>	UPPER RIGHT CORNER
U+0231F	└	<code>\lrcorner</code>	LOWER RIGHT CORNER
U+02773		<code>\rbrbrak</code>	ORNAMENT
U+027C6	⎵	<code>\rbag</code>	RIGHT S-SHAPED BAG DELIMITER
U+027E7	⎶	<code>\rBrack</code>	BRACKET
U+027E9	⋗	<code>\rangle</code>	MATHEMATICAL RIGHT ANGLE BRACKET
U+027EB	⋘	<code>\rAngle</code>	BRACKET
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	⎻	<code>\rbrackubar</code>	RIGHT SQUARE BRACKET WITH UNDERBAR
U+0298E	⎼	<code>\rbracklrtick</code>	BOTTOM CORNER

			RIGHT SQUARE BRACKET WITH TICK IN TOP
U+02990		<code>\rbrackurtick</code>	CORNER
U+02992		<code>\rangledot</code>	RIGHT ANGLE BRACKET WITH DOT
U+02994		<code>\rparengtr</code>	RIGHT ARC GREATER-THAN BRACKET
U+02998		<code>\rblkbbrbrak</code>	RIGHT BLACK TORTOISE SHELL BRACKET
U+029D9		<code>\rvzigzag</code>	RIGHT WIGGLY FENCE
U+029DB		<code>\Rvzigzag</code>	RIGHT DOUBLE WIGGLY FENCE
U+029FD		<code>\rcurvyangle</code>	RIGHT POINTING CURVED ANGLE BRACKET
U+03015		<code>\rbrbrak</code>	RIGHT BROKEN BRACKET
U+03019		<code>\Rbrbrak</code>	RIGHT WHITE TORTOISE SHELL BRACKET

5.7 Maths accents

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

USV	Ex.	Macro	Description
U+00300		<code>\grave</code>	GRAVE ACCENT
U+00301		<code>\acute</code>	ACUTE ACCENT
U+00302		<code>\hat</code>	CIRCUMFLEX ACCENT
U+00303		<code>\tilde</code>	TILDE
U+00304		<code>\bar</code>	MACRON
U+00305		<code>\overbar</code>	OVERBAR EMBELLISHMENT
U+00306		<code>\breve</code>	BREVE
U+00307		<code>\dot</code>	DOT ABOVE
U+00308		<code>\ddot</code>	DIERESIS
U+00309		<code>\ovhook</code>	COMBINING HOOK ABOVE
U+0030A		<code>\ocirc</code>	RING
U+0030C		<code>\check</code>	CARON
U+00310		<code>\candra</code>	CANDRABINDU (NON-SPACING)
U+00312		<code>\oturnedcomma</code>	COMBINING TURNED COMMA ABOVE
			GREEK PSILI (SMOOTH BREATHING)
U+00313		<code>\osmooth</code>	(NON-SPACING)
			GREEK DASIA (ROUGH BREATHING)
U+00314		<code>\orough</code>	(NON-SPACING)
U+00315		<code>\ocommatopright</code>	COMBINING COMMA ABOVE RIGHT
U+0031A		<code>\droang</code>	LEFT ANGLE ABOVE (NON-SPACING)
			COMBINING LONG SOLIDUS OVERLAY
U+00338		<code>\not</code>	NOT
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>\overrightarrow</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>\overleftrightharpoonarrow</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
U+020EC		<code>\underrightharpoondown</code>	BARB DOWNWARDS COMBINING LEFTWARDS HARPOON WITH
U+020ED		<code>\underleftharpoondown</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

6 Font features

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

```

632 \newcommand\um@zf@feature[2]{
633   \define@key[zf]{options}{#1}[]{}
634   \if@um@fontspec@feature
635     #2
636   \else
637     \PackageError{fontspec/unicode-math}
638       {The ‘#1’ font feature can only be used for maths fonts}
639       {The feature you tried to use can only be in commands
640        like \protect\setmathfont}
641   \fi
642 }
643 }
```

6.1 OpenType maths font features

```

644 \um@zf@feature{ScriptStyle}{
645   \zf@update@ff{+ssty=0}
646 }
647 \um@zf@feature{ScriptScriptStyle}{
648   \zf@update@ff{+ssty=1}
649 }
```

6.2 Script and scriptscript font options

```

650 \define@cmdkey[um]{options}[um@]{ScriptFeatures}{}
651 \define@cmdkey[um]{options}[um@]{ScriptScriptFeatures}{}

```

```

652 \define@cmdkey[um]{options}[um@]{ScriptFont}{}
653 \define@cmdkey[um]{options}[um@]{ScriptScriptFont}{}

```

6.3 Range processing

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

```

654 \define@choicekey+[um]{options}{Range}[\@tempa\@tempb]{ALL}{
655   \ifcase\@tempb\relax
656     \global\let\um@char@range\@empty
657   \fi
658 }{
659   \xdef\um@char@range{#1}
660 }

```

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

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

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

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

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

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

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

```

661 \newcommand\um@parse@term[4]{
662   \clist_map_variable:NNn \um@char@range \@ii {
663     \unless\ifx\@ii\@empty
664       \@tempswafalse

```

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

```

665     \expandafter\um@firstchar\expandafter{\@ii}
666     \ifx\@tempa\um@backslash

```

```

667     \expandafter\ifx\@ii#2\relax
668         \@tempswatrue
669     \else
670         \expandafter\ifx\@ii#3\relax
671             \@tempswatrue
672         \fi
673     \fi

```

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

```

674     \else
675         \expandafter\um@parse@range\@ii-\@marker-\@nil#1\@nil
676     \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.

```

677     \if@tempswa
678         \ifx\um@char@num@range\@empty
679             \g@addto@macro\um@char@num@range{#1}
680         \else
681             \g@addto@macro\um@char@num@range{, #1}
682         \fi
683         #4%
684     \fi
685 \fi
686 }
687 }
688 \def\um@firstof#1#2\@nil{#1}
689 \edef\um@backslash{\expandafter\um@firstof\string\string\@nil}
690 \def\um@firstchar#1{\edef\@tempa{\expandafter\um@firstof\string#1\@nil}}

```

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

```

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

```

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

```

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

```

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


```

694 \expandafter\ifx\expandafter\@marker\@tempb\relax
695 \ifnum#4=#1\relax
696 \@tempwattrue
697 \fi
698 \else


---


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


---


699 \ifx\@empty\@tempb
700 \ifnum#4>\numexpr#1-1\relax
701 \@tempwattrue
702 \fi
703 \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-


---


704 \ifx\@empty\@tempa
705 \ifnum#4<\numexpr#2+1\relax
706 \@tempwattrue
707 \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-


---


708 \else
709 \ifnum#4>\numexpr#1-1\relax
710 \ifnum#4<\numexpr#2+1\relax
711 \@tempwattrue
712 \fi
713 \fi
714 \fi
715 \fi
716 \fi
717 }

```

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

```

718 \cs_set:Nn \um_map_chars_range:nnn {
719 \clist_map_variable:nNn {#2} \l_um_input_num {
720 \prg_stepwise_variable:nnnNn{0}{1}{#1} \l_um_incr_num {

```

```

721     \um_set_mathcode:nnnn
722     {\numexpr \l_um_incr_num+ \l_um_input_num \relax}
723     {\mathalpha}{\um_symfont_t1}
724     {\numexpr \l_um_incr_num + #3 \relax}
725   }
726 }
727 }
728 \cs_set:Nn \um_map_chars_latin:nn {
729   \um_map_chars_range:nnn {25}{#1}{#2}
730 }
731 \cs_set:Nn \um_map_chars_greek:nn {
732   \um_map_chars_range:nnn {24}{#1}{#2}
733 }
734 \cs_set:Nn \um_map_chars_numbers:nn {
735   \um_map_chars_range:nnn {9}{#1}{#2}
736 }
737 \cs_set:Nn \um_map_char:nn {
738   \um_map_chars_range:nnn {0}{#1}{#2}
739 }

\um_set_mathalphabet_char:Nnnn #1 : Maths alphabet
#2 : Input char(s)
#3 : Output char
Loops through character ranges setting \mathcode.
740 \cs_set:Npn \exp_args:Nnff {\::n\::f\::f\:::}
741 \cs_new:Nn \um_set_mathalphabet_char:Nnn {
742   \clist_map_variable:nNn {#2} \l_um_input_num {
743     \exp_args:Nnff \um_mathmap:Nnn {#1}
744     {\number\numexpr\l_um_input_num\relax} {\number\numexpr#3\relax}
745   }
746 }

\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.
747 \cs_new:Nn \um_set_mathalph_range:Nnn {
748   \clist_map_variable:nNn {#3} \l_um_input_num {
749     \prg_stepwise_variable:nnnNn {0}{1}{#1} \l_um_inc_num {
750       \exp_args:Nnff \um_mathmap:Nnn {#2}
751       {\number\numexpr \l_um_inc_num + \l_um_input_num \relax}
752       {\number\numexpr \l_um_inc_num + #4 \relax}
753     }
754   }
755 }
756 \cs_new:Nn \um_set_mathalphabet_numbers:Nnn {

```

```

757 \um_set_mathalph_range:nNnn {9}{#1}{#2}{#3}
758 }
759 \cs_new:Nn \um_set_mathalphabet_latin:Nnn {
760 \um_set_mathalph_range:nNnn {25}{#1}{#2}{#3}
761 }
762 \cs_new:Nn \um_set_mathalphabet_greek:Nnn {
763 \um_set_mathalph_range:nNnn {24}{#1}{#2}{#3}
764 }

```

BCDBCDEABCDEF

```

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

```

6.4 Resolving Greek symbol name control sequences

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

```

765 \AtBeginDocument{\um@resolve@greek}
766 \newcommand\um@resolve@greek{
767 \def\Alpha{\mitAlpha}
768 \def\Beta{\mitBeta}
769 \def\Gamma{\mitGamma}
770 \def\Delta{\mitDelta}
771 \def\Epsilon{\mitEpsilon}
772 \def\Zeta{\mitZeta}
773 \def\Eta{\mitEta}
774 \def\Theta{\mitTheta}
775 \def\Iota{\mitIota}
776 \def\Kappa{\mitKappa}
777 \def\Lambda{\mitLambda}
778 \def\Mu{\mitMu}
779 \def\Nu{\mitNu}
780 \def\Xi{\mitXi}
781 \def\Omicron{\mitOmicron}
782 \def\Pi{\mitPi}
783 \def\Rho{\mitRho}
784 \def\varTheta{\mitvarTheta}
785 \def\Sigma{\mitSigma}
786 \def\Tau{\mitTau}
787 \def\Upsilon{\mitUpsilon}
788 \def\Phi{\mitPhi}
789 \def\Chi{\mitChi}

```

```

790 \def\Psi{\mitPsi}
791 \def\Omega{\mitOmega}

Lowercase:

792 \def\alpha{\mitalpha}
793 \def\beta{\mitbeta}
794 \def\gamma{\mitgamma}
795 \def\delta{\mitdelta}
796 \def\epsilon{
797   \bool_if:NTF \g_um_texgreek_bool {\mitvarepsilon}{\mitepsilon}
798 }
799 \def\zeta{\mitzeta}
800 \def\eta{\miteta}
801 \def\theta{\mittheta}
802 \def\iota{\mitiota}
803 \def\kappa{\mitkappa}
804 \def\lambda{\mitlambda}
805 \def\mu{\mitmu}
806 \def\nu{\mitnu}
807 \def\xi{\mitxi}
808 \def\omicron{\mitomicron}
809 \def\pi{\mitpi}
810 \def\rho{\mitrho}
811 \def\varsigma{\mitvarsigma}
812 \def\sigma{\mitsigma}
813 \def\tau{\mittau}
814 \def\upsilon{\mitupsilon}
815 \def\phi{
816   \bool_if:NTF \g_um_texgreek_bool {\mitvarphi}{\mitphi}
817 }
818 \def\chi{\mitchi}
819 \def\psi{\mitpsi}
820 \def\omega{\mitomega}
821 \def\varepsilon{
822   \bool_if:NTF \g_um_texgreek_bool {\mitepsilon}{\mitvarepsilon}
823 }
824 \def\vartheta{\mitvartheta}
825 \def\varkappa{\mitvarkappa}
826 \def\varphi{
827   \bool_if:NTF \g_um_texgreek_bool {\mitphi}{\mitvarphi}
828 }
829 \def\varrho{\mitvarrho}
830 \def\varpi{\mitvarpi}
831 }

```

6.5 Setting up the mappings

`\um_setup_literals:` : TODO : other literal symbols

```
832 \cs_set:Nn \um_setup_literals: {  
833   \um_map_chars_latin:nn {\um@usv@upLatin}{\um@usv@upLatin}  
834   \um_map_chars_latin:nn {\um@usv@itLatin}{\um@usv@itLatin}  
835   \um_map_chars_latin:nn {\um@usv@itlatin}{\um@usv@itlatin}  
836   \um_map_char:nn {\um@usv@ith}{\um@usv@ith}  
837   \um_map_chars_latin:nn {\um@usv@uplatin}{\um@usv@uplatin}  
838   \um_map_chars_greek:nn {\um@usv@upGreek}{\um@usv@upGreek}  
839   \um_map_char:nn {\um@usv@varTheta}{\um@usv@varTheta}  
840   \um_map_chars_greek:nn {\um@usv@itGreek}{\um@usv@itGreek}  
841   \um_map_chars_greek:nn {\um@usv@upgreek}{\um@usv@upgreek}  
842 }
```

`\um_setup_bf_literals:` TODO: other literal symbols

```
843 \cs_set:Nn \um_setup_bf_literals: {  
844   \um_map_chars_latin:nn {\um@usv@bfupLatin}{\um@usv@bfupLatin}  
845   \um_map_chars_latin:nn {\um@usv@bfuplatin}{\um@usv@bfuplatin}  
846   \um_map_chars_latin:nn {\um@usv@bfitLatin}{\um@usv@bfitLatin}  
847   \um_map_chars_latin:nn {\um@usv@bfitlatin}{\um@usv@bfitlatin}  
848   \um_map_chars_greek:nn {\um@usv@bfupGreek}{\um@usv@bfupGreek}  
849   \um_map_chars_greek:nn {\um@usv@bfupgreek}{\um@usv@bfupgreek}  
850   \um_map_chars_greek:nn {\um@usv@bfitGreek}{\um@usv@bfitGreek}  
851   \um_map_chars_greek:nn {\um@usv@bfitgreek}{\um@usv@bfitgreek}  
852 }
```

`\um_setup_Latin:`

```
853 \cs_set:Nn \um_setup_Latin: {  
854   \if@um@upLatin  
855     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}  
856   \else  
857     \um_map_chars_latin:nn {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}  
858   \fi  
859 }
```

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

```
860 \cs_set:Nn \um_setup_latin: {  
861   \if@um@uplatin  
862     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}  
863     \um_map_char:nn {\um@usv@ith}{`\h}  
864   \else  
865     \um_map_chars_latin:nn {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}  
866     \um_map_char:nn {\h,\um@usv@ith}{\um@usv@ith}  
867   \fi  
868 }
```

`\um_setup_Greek:`

```
869 \cs_set:Nn \um_setup_Greek: {  
870   \if@um@upGreek  
871     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}  
872     \um_map_char:nn {\um@usv@varTheta,"1D6F3}{\um@usv@varTheta}  
873   \else  
874     \um_map_chars_greek:nn {\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}  
875     \um_map_char:nn {\um@usv@varTheta}{\um@usv@itvarTheta}  
876   \fi  
877 }
```

`\um_setup_greek:`

```
878 \cs_set:Nn \um_setup_greek: {  
879   \if@um@upgreek  
880     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}  
881     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}  
882     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}  
883     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}  
884     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}  
885     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}  
886     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}  
887   \else  
888     \um_map_chars_greek:nn {\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}  
889     \um_map_char:nn {\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}  
890     \um_map_char:nn {\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}  
891     \um_map_char:nn {\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}  
892     \um_map_char:nn {\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}  
893     \um_map_char:nn {\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}  
894     \um_map_char:nn {\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}  
895   \fi  
896 }
```

7 Maths alphabets mapping definitions

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

Implicit mode:

- Try and set all of the alphabet shapes.
- Check for the first glyph of the uppercase Latin alphabet to detect if the font supports each alphabet shape. (This doesn't work to distinguish Latin/Greek but we hope all maths fonts will have at least them!)
- For alphabets that do exist, overwrite whatever's already there.

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

Explicit mode:

- Only set the alphabets specified.
- Check for the first glyph of the uppercase Latin alphabet to detect if the font contains the alphabet shape in the unicode math plane.
- For unicode math alphabets, overwrite whatever's already there.
- Otherwise, use the ASCII letters instead.

```

897 \cs_new:Nn \um_setup_alphabets: {
898   \um_setup_math_alphabet:nn {up}    {{latin, Latin, greek, Greek}}
899   \um_setup_math_alphabet:nn {it}    {{latin, Latin, greek, Greek}}
900   \um_setup_math_alphabet:nn {bb}    {{latin, Latin, num}}
901   \um_setup_math_alphabet:nn {scr}   {{latin, Latin}}
902   \um_setup_math_alphabet:nn {frak}  {{latin, Latin}}
903   \um_setup_math_alphabet:nn {sf}    {{latin, Latin, num}}
904   \um_setup_math_alphabet:nn {sfup}  {{latin, Latin, num}}
905   \um_setup_math_alphabet:nn {sfit}  {{latin, Latin, num}}
906   \um_setup_math_alphabet:nn {tt}    {{latin, Latin, num}}
907   \um_setup_math_alphabet:nn {bf}    {{latin, Latin, greek, Greek, num}}
908   \um_setup_math_alphabet:nn {bfup}  {{latin, Latin, greek, Greek, num}}
909   \um_setup_math_alphabet:nn {bfit}  {{latin, Latin, greek, Greek, num}}
910   \um_setup_math_alphabet:nn {bfscr} {{latin, Latin}}
911   \um_setup_math_alphabet:nn {bffrak}{{latin, Latin}}
912   \um_setup_math_alphabet:nn {bfsf}  {{latin, Latin, greek, Greek, num}}
913   \um_setup_math_alphabet:nn {bfsfup}{{latin, Latin, greek, Greek, num}}
914   \um_setup_math_alphabet:nn {bfsfit}{{latin, Latin, greek, Greek, num}}
915 }

```

`\um_setup_math_alphabet:nn` #1 : Math font family name (e.g., 'sf')

#2 : Math alphabets, comma separated of {latin, Latin, greek, Greek, num}

First check that at least one of the alphabets for the font shape is defined, and then then loop through them defining the individual ranges.

```

916 \cs_new:Nn \um_setup_math_alphabet:nn {
917   \clist_map_inline:nn {#2} {
918     \um_glyph_if_exist:nT {\csname um@usv@#1##1 \endcsname}{
919       \um_maybe_init_alphabet:n {#1}
920       \um_prepare_alph:n {#1}
921       \clist_map_break:
922     }
923   }
924   \clist_map_inline:nn {#2} {
925     \um_glyph_if_exist:nTF {\csname um@usv@#1##1 \endcsname}{

```

```

926     \use:c {um_config_math#1_##1:}
927   }{
928     \PackageWarningNoLine{unicode-math}{^^J\space\space\space\space
929     Math~ alphabet~
930     \@backslashchar math#1~
931     (\tl_use:c{g_um_math_alphabet_name_##1_tl})~
932     not~ found~ in~ font~
933     \fontname\um@font}
934   }
935 }
936 }
937 \tl_set:Nn \g_um_math_alphabet_name_latin_tl {Latin, lowercase}
938 \tl_set:Nn \g_um_math_alphabet_name_Latin_tl {Latin, uppercase}
939 \tl_set:Nn \g_um_math_alphabet_name_greek_tl {Greek, lowercase}
940 \tl_set:Nn \g_um_math_alphabet_name_Greek_tl {Greek, uppercase}
941 \tl_set:Nn \g_um_math_alphabet_name_num_tl {Numerals}

942 \cs_set:Nn \um_init_alphabet:n {
943   \cs_set_eq:cN {um_setup_math#1:} \prg_do_nothing:
944 }

```

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

```

945 \prg_new_conditional:Nnn \um_glyph_if_exist:n {p,TF,T,F} {
946   \etex_iffontchar:D \um@font #1 \scan_stop: \prg_return_true: \else: \prg_return_false: \fi:
947 }

```

`\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.

```

948 \cs_new:Nn \um_prepare_alph:n {
949   \cs_if_exist:cF {um_math#1:n} {
950     \cs_set:cpn {um_math#1:n} ##1 {
951       \use:c {um_setup_math#1:} ##1 \egroup
952     }
953     \cs_set_protected:cpn {math#1} {
954       \bgroup
955       \mode_if_math:F {
956         \egroup\expandafter
957         \non@alpherr\expandafter{\csname math#1\endcsname\space}
958       }
959       \use:c {um_math#1:n}
960     }
961   }
962 }

```

: TODO : nested alphabets?

7.1 Non-bold math alphabets

7.1.1 Upright: `\mathup`

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

```
963 \cs_new:Npn \um_config_mathup_Latin: {
964   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@upLatin}
965 }
966 \cs_new:Npn \um_config_mathup_latin: {
967   \um_set_mathalphabet_latin:Nnn{\mathup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@uplatin}
968   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@ith} {\`h}
969 }
970 \cs_new:Npn \um_config_mathup_Greek: {
971   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@upGreek}
972   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@Nabla}
973   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@varTheta}
974 }
975 \cs_new:Npn \um_config_mathup_greek: {
976   \um_set_mathalphabet_greek:Nnn{\mathup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@upgreek}
977   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@partial}
978   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@varepsilon}
979   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@vartheta}
980   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@varkappa}
981   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@varphi}
982   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@varrho}
983   \um_set_mathalphabet_char:Nnn{\mathup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@varpi}
984 }
```

7.1.2 Italic: `\mathit`

```
985 \cs_new:Npn \um_config_mathit_Latin: {
986   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@itLatin}
987 }
988 \cs_new:Npn \um_config_mathit_latin: {
989   \um_set_mathalphabet_latin:Nnn{\mathit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@itlatin}
990   \um_set_mathalphabet_char:Nnn{\mathit}{\`h,\um@usv@ith}{\um@usv@ith}
991 }
992 \cs_new:Npn \um_config_mathit_Greek: {
993   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@itGreek}
994   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@itvarTheta}
995   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@itNabla}
996 }
997 \cs_new:Npn \um_config_mathit_greek: {
998   \um_set_mathalphabet_greek:Nnn{\mathit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@itgreek}
999   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@itpartial}
1000   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@itvarepsilon}
1001   \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@itvartheta}
```

```

1002 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@itvarkappa}
1003 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@itvarphi}
1004 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@itvarrho}
1005 \um_set_mathalphabet_char:Nnn{\mathit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@itvarpi}
1006 }

```

7.1.3 Blackboard or double-struck: `\mathbb`

```

1007 \cs_new:Npn \um_config_mathbb_latin: {
1008   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bblatin}
1009   \um_set_mathalphabet_char:Nnn{\mathbb}{\um@usv@ith} {"1D559}
1010 }
1011 \cs_new:Npn \um_config_mathbb_Latin: {
1012   \um_set_mathalphabet_latin:Nnn{\mathbb}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bbLatin}
1013   \um_set_mathalphabet_char:Nnn{\mathbb}{`C,"1D436}{ "2102}
1014   \um_set_mathalphabet_char:Nnn{\mathbb}{`H,"1D43B}{ "210D}
1015   \um_set_mathalphabet_char:Nnn{\mathbb}{`N,"1D441}{ "2115}
1016   \um_set_mathalphabet_char:Nnn{\mathbb}{`P,"1D443}{ "2119}
1017   \um_set_mathalphabet_char:Nnn{\mathbb}{`Q,"1D444}{ "211A}
1018   \um_set_mathalphabet_char:Nnn{\mathbb}{`R,"1D445}{ "211D}
1019   \um_set_mathalphabet_char:Nnn{\mathbb}{`Z,"1D44D} {"2124}
1020 }
1021 \cs_new:Npn \um_config_mathbb_num: {
1022   \um_set_mathalphabet_numbers:Nnn{\mathbb}{\um@usv@num}{\um@usv@bnum}
1023 }

```

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

```

1024 \cs_new:Npn \um_config_mathscr_Latin: {
1025   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@scrLatin}
1026   \um_set_mathalphabet_char:Nnn \mathscr {\`B,"1D435}{ "212C}
1027   \um_set_mathalphabet_char:Nnn \mathscr {\`E,"1D438}{ "2130}
1028   \um_set_mathalphabet_char:Nnn \mathscr {\`F,"1D439}{ "2131}
1029   \um_set_mathalphabet_char:Nnn \mathscr {\`H,"1D43B}{ "210B}
1030   \um_set_mathalphabet_char:Nnn \mathscr {\`I,"1D43C}{ "2110}
1031   \um_set_mathalphabet_char:Nnn \mathscr {\`L,"1D43F}{ "2112}
1032   \um_set_mathalphabet_char:Nnn \mathscr {\`M,"1D440}{ "2133}
1033   \um_set_mathalphabet_char:Nnn \mathscr {\`R,"1D445}{ "211B}
1034 }
1035 \cs_new:Npn \um_config_mathscr_latin: {
1036   \um_set_mathalphabet_latin:Nnn \mathscr {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@scrlatin}
1037   \um_set_mathalphabet_char:Nnn \mathscr {\`e,"1D452}{ "212F}
1038   \um_set_mathalphabet_char:Nnn \mathscr {\`g,"1D454}{ "210A}
1039   \um_set_mathalphabet_char:Nnn \mathscr {\`o,"1D45C}{ "2134}
1040   \um_set_mathalphabet_char:Nnn \mathscr {\um@usv@ith} {"1D4BD}
1041 }

```

7.1.5 Fraktur or fraktur or blackletter: `\mathfrak`

```

1042 \cs_new:Npn \um_config_mathfrak_Latin: {
1043   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@upLatin,\um@usv@itLatin}{\um@usv@frakLat
1044   \um_set_mathalphabet_char:Nnn \mathfrak {\`C,"1D436}{ "212D}
1045   \um_set_mathalphabet_char:Nnn \mathfrak {\`H,"1D43B}{ "210C}
1046   \um_set_mathalphabet_char:Nnn \mathfrak {\`I,"1D43C}{ "2111}
1047   \um_set_mathalphabet_char:Nnn \mathfrak {\`R,"1D445}{ "211C}
1048   \um_set_mathalphabet_char:Nnn \mathfrak {\`Z,"1D44D}{ "2128}
1049 }
1050 \cs_new:Npn \um_config_mathfrak_latin: {
1051   \um_set_mathalphabet_latin:Nnn \mathfrak {\um@usv@uplatin,\um@usv@itlatin}{\um@usv@fraklati
1052   \um_set_mathalphabet_char:Nnn \mathfrak {\um@usv@ith} {"1D525}
1053 }

```

7.1.6 Sans serif: \mathsf

```

1054 \cs_new:Npn \um_config_mathsf_Latin: {
1055   \bool_if:NTF \g_um_sfliteral_bool {
1056     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@upLatin}{\um@usv@sfupLatin}
1057     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itLatin}{\um@usv@sfitLatin}
1058   }{
1059     \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@upLatin,\um@usv@itLatin}{ \um_sf_Latin_up
1060   }
1061 }
1062 \cs_new:Npn \um_config_mathsf_latin: {
1063   \bool_if:NTF \g_um_sfliteral_bool {
1064     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@uplatin}{\um@usv@sfuplatin}
1065     \um_set_mathalphabet_latin:Nnn{\mathsf}{\um@usv@itlatin}{\um@usv@sfitlatin}
1066     \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}
1067   }{
1068     \um_set_mathalphabet_latin:Nnn \mathsf {\um@usv@uplatin,\um@usv@itlatin}{ \um_sf_latin_up
1069     \bool_if:NTF \g_um_upsans_bool {
1070       \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D5C1}
1071     }{
1072       \um_set_mathalphabet_char:Nnn \mathsf {\um@usv@ith} {"1D629}
1073     }
1074   }
1075 }
1076 \cs_new:Npn \um_config_mathsf_num: {
1077   \um_set_mathalphabet_numbers:Nnn{\mathsf}{\um@usv@num}{\um@usv@sfnum}
1078 }

```

7.1.7 Sans serif upright: \mathsfup

```

1079 \cs_new:Npn \um_config_mathsfup_num: {
1080   \um_set_mathalphabet_numbers:Nnn{\mathsfup}{\um@usv@num}{\um@usv@sfnum}
1081 }
1082 \cs_new:Npn \um_config_mathsfup_latin: {
1083   \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfuplat
1084   \um_set_mathalphabet_char:Nnn \mathsfup {\um@usv@ith} {"1D5C1}

```

```

1085 }
1086 \cs_new:Npn \um_config_mathsfup_Latin: {
1087   \um_set_mathalphabet_latin:Nnn{\mathsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfupLat
1088 }

```

7.1.8 Sans serif italic: `\mathsfit`

Map the numbers like that because it seems sensible.

```

1089 \cs_new:Npn \um_config_mathsfnum: {
1090   \um_set_mathalphabet_numbers:Nnn{\mathsfnum}{\um@usv@num}{\um@usv@sfnum}
1091 }
1092 \cs_new:Npn \um_config_mathsfit_Latin: {
1093   \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@sfitLat
1094 }
1095 \cs_new:Npn \um_config_mathsfit_latin: {
1096   \um_set_mathalphabet_latin:Nnn{\mathsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@sfitlat
1097   \um_set_mathalphabet_char:Nnn \mathsfit {\um@usv@ith} {"1D629}
1098 }

```

7.1.9 Typewriter or monospaced: `\mathtt`

```

1099 \cs_new:Npn \um_config_mathtt_num: {
1100   \um_set_mathalphabet_numbers:Nnn{\mathtt}{\um@usv@num}{\um@usv@ttnum}
1101 }
1102 \cs_new:Npn \um_config_mathtt_Latin: {
1103   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@ttLatin}
1104 }
1105 \cs_new:Npn \um_config_mathtt_latin: {
1106   \um_set_mathalphabet_latin:Nnn{\mathtt}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@ttlatin}
1107   \um_set_mathalphabet_char:Nnn \mathtt {\um@usv@ith} {"1D691}
1108 }

```

7.2 Bold math alphabets

7.2.1 Bold: `\mathbf`

```

1109 \cs_new:Npn \um_config_mathbf_num: {
1110   \um_set_mathalphabet_numbers:Nnn{\mathbf}{\um@usv@num}{\um@usv@bfnum}
1111 }
1112 \cs_new:Npn \um_config_mathbf_Latin: {
1113   \if@um@bfliteral
1114     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin}{\um@usv@bfupLatin}
1115     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itLatin}{\um@usv@bfitLatin}
1116   \else
1117     \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bf_Latin_up_
1118     \fi
1119 }
1120 \cs_new:Npn \um_config_mathbf_latin: {

```

```

1121 \if@um@bfliteral
1122 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin}{\um@usv@bfuplatin}
1123 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@itlatin}{\um@usv@bfitlatin}
1124 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1125 \else
1126 \um_set_mathalphabet_latin:Nnn{\mathbf}{\um@usv@uplatin,\um@usv@itlatin}{\um_bf_latin_up_
1127 \if@um@bfuplatin
1128 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfuph}
1129 \else
1130 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@ith}{\um@usv@bfith}
1131 \fi
1132 \fi
1133 }
1134 \cs_new:Npn \um_config_mathbf_Greek: {
1135 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
1136 \if@um@bfliteral
1137 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek}{\um@usv@bfupGreek}
1138 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itGreek}{\um@usv@bfitGreek}
1139 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta}{\um@usv@bfvarTheta}
1140 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
1141 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla}{\um@usv@bfNabla}
1142 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itNabla}{\um@usv@bfitNabla}
1143 \else
1144 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upGreek,\um@usv@itGreek}{\um_bf_Greek_up_
1145 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Nabla,\um@usv@itNabla}{\um_bfNabla_up_or_i
1146 \if@um@bfupGreek
1147 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfv
1148 \else
1149 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfv
1150 \fi
1151 \fi
1152 }
1153 \cs_new:Npn \um_config_mathbf_greek: {
1154 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
1155 \if@um@bfliteral
1156 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek}{\um@usv@bfupgreek}
1157 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@itgreek}{\um@usv@bfitgreek}
1158 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial}{\um@usv@bfpartial}
1159 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon}{\um@usv@bfvarepsilon}
1160 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta}{\um@usv@bfvartheta}
1161 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa}{\um@usv@bfvarkappa}
1162 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi}{\um@usv@bfvarphi}
1163 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho}{\um@usv@bfvarrho}
1164 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi}{\um@usv@bfvarpi}
1165 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itpartial}{\um@usv@bfitpartial}
1166 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarepsilon}{\um@usv@bfitvarepsilon}

```

```

1167 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvartheta}{\um@usv@bfitvartheta}
1168 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarkappa}{\um@usv@bfitvarkappa}
1169 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarphi}{\um@usv@bfitvarphi}
1170 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarrho}{\um@usv@bfitvarrho}
1171 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@itvarpi}{\um@usv@bfitvarpi}
1172 \else
1173 \um_set_mathalphabet_greek:Nnn{\mathbf}{\um@usv@upgreek,\um@usv@itgreek}{\um_bf_greek_up}
1174 \if@um@bfupgreek
1175 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
1176 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1177 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1178 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1179 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1180 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1181 \else
1182 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@bfvarepsilon}
1183 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfvartheta}
1184 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfvarkappa}
1185 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi}
1186 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho}
1187 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1188 \fi
1189 \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@partial,\um@usv@itpartial}{\um_bfpartial_u}
1190 \fi
1191 }

```

7.2.2 Bold Italic: `\mathbfit`

```

1192 \cs_new:Npn \um_config_mathbfit_num: {
1193   \um_set_mathalphabet_numbers:Nnn{\mathbfit}{\um@usv@num}{\um@usv@bfnum}
1194 }
1195 \cs_new:Npn \um_config_mathbfit_Latin: {
1196   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1197 }
1198 \cs_new:Npn \um_config_mathbfit_latin: {
1199   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfitlatin}
1200   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@ith} {"1D489}
1201 }
1202 \cs_new:Npn \um_config_mathbfit_Greek: {
1203   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfitGreek}
1204   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfitvarTheta}
1205   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfitNabla}
1206 }
1207 \cs_new:Npn \um_config_mathbfit_greek: {
1208   \um_set_mathalphabet_latin:Nnn{\mathbfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfitLatin}
1209   \um_set_mathalphabet_greek:Nnn{\mathbfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfitgreek}
1210   \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfitpartial}

```

```

1211 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
1212 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfit
1213 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfit
1214 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfitvarp
1215 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfitvarr
1216 \um_set_mathalphabet_char:Nnn{\mathbfit}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfitvarpi}
1217 }

```

7.2.3 Bold Italic: \mathbf{fup}

```

1218 \cs_new:Npn \um_config_mathbfup_num: {
1219   \um_set_mathalphabet_numbers:Nnn{\mathbfup}{\um@usv@num}{\um@usv@bfnum}
1220 }
1221 \cs_new:Npn \um_config_mathbfup_Latin: {
1222   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfupLat
1223 }
1224 \cs_new:Npn \um_config_mathbfup_latin: {
1225   \um_set_mathalphabet_latin:Nnn{\mathbfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfuplat
1226   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@ith} {"1D421}
1227 }
1228 \cs_new:Npn \um_config_mathbfup_Greek: {
1229   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfupGre
1230   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varTheta,\um@usv@itvarTheta}{\um@usv@bfva
1231   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfNabla}
1232   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@Digamma}{\um@usv@bfDigamma}
1233 }
1234 \cs_new:Npn \um_config_mathbfup_greek: {
1235   \um_set_mathalphabet_greek:Nnn{\mathbfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfupgre
1236   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfpart
1237   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{\um@usv@
1238   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@vartheta,\um@usv@itvartheta}{\um@usv@bfva
1239   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varkappa,\um@usv@itvarkappa}{\um@usv@bfva
1240   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varphi,\um@usv@itvarphi}{\um@usv@bfvarphi
1241   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varrho,\um@usv@itvarrho}{\um@usv@bfvarrho
1242   \um_set_mathalphabet_char:Nnn{\mathbfup}{\um@usv@varpi,\um@usv@itvarpi}{\um@usv@bfvarpi}
1243   \um_set_mathalphabet_char:Nnn{\mathbf}{\um@usv@digamma}{\um@usv@bfdigamma}
1244 }

```

7.2.4 Bold fractur or fraktur or blackletter: \mathbf{ffrak}

```

1245 \cs_new:Npn \um_config_mathbffrak_Latin: {
1246   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bffra
1247 }
1248 \cs_new:Npn \um_config_mathbffrak_latin: {
1249   \um_set_mathalphabet_latin:Nnn{\mathbffrak}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bffra
1250   \um_set_mathalphabet_char:Nnn{\mathbffrak}{\um@usv@ith} {"1D58D}
1251 }

```

7.2.5 Bold script or calligraphic: `\mathbfscr`

```

1252 \cs_new:Npn \um_config_mathbfscr_Latin: {
1253   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfscrL
1254 }
1255 \cs_new:Npn \um_config_mathbfscr_latin: {
1256   \um_set_mathalphabet_latin:Nnn{\mathbfscr}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfscrL
1257   \um_set_mathalphabet_char:Nnn{\mathbfscr}{\um@usv@ith} {"1D4F1}
1258 }

```

7.2.6 Bold sans serif: `\mathbfsf`

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

```

1259 \cs_new:Npn \um_config_mathbfsf_num: {
1260   \um_set_mathalphabet_numbers:Nnn \mathbfsf {\um@usv@num}{\um@usv@bfsfnum}
1261 }
1262 \cs_new:Npn \um_config_mathbfsf_Latin: {
1263   \bool_if:NTF \g_um_sfliteral_bool {
1264     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin}{\um@usv@bfsfupLatin}
1265     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itLatin}{\um@usv@bfsfitLatin}
1266   }{
1267     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um_bfsf_Lati
1268   }
1269 }
1270 \cs_new:Npn \um_config_mathbfsf_latin: {
1271   \bool_if:NTF \g_um_sfliteral_bool {
1272     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@uplatin}{\um@usv@bfsfuplatin}
1273     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@itlatin}{\um@usv@bfsfitlatin}
1274     \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
1275   }{
1276     \um_set_mathalphabet_latin:Nnn \mathbfsf {\um@usv@upLatin,\um@usv@itLatin}{\um_bfsf_lati
1277     \bool_if:NTF \g_um_upsans_bool {
1278       \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D5F5}
1279     }{
1280       \um_set_mathalphabet_char:Nnn \mathbfsf{\um@usv@ith} {"1D65D}
1281     }
1282   }
1283 }
1284 \cs_new:Npn \um_config_mathbfsf_Greek: {
1285   \bool_if:NTF \g_um_sfliteral_bool {
1286     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek}{\um@usv@bfsfupGreek}
1287     \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itGreek}{\um@usv@bfsfitGreek}
1288     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{{"1D767}
1289     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta}{{"1D7A1}
1290     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla}{\um@usv@bfsfNabla}
1291     \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itNabla}{\um@usv@bfsfitNabla}
1292   }{

```



```

1293 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upGreek,\um@usv@itGreek}{\um_bfsf_Gree
1294 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@Nabla,\um@usv@itNabla}{\um_bfsfNabla_up
1295 \bool_if:NTF \g_um_upsans_bool {
1296 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{1D767}
1297 }{
1298 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varTheta,\um@usv@itvarTheta}{1D7A1}
1299 }
1300 }
1301 }
1302 \cs_new:Npn \um_config_mathbfsf_greek: {
1303 \bool_if:NTF \g_um_sfliteral_bool {
1304 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek}{\um@usv@bfsfupgreek}
1305 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@itgreek}{\um@usv@bfsfitgreek}
1306 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial}{\um@usv@bfsfpartial}
1307 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon}{1D78A}
1308 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta}{1D78B}
1309 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa}{1D78C}
1310 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi}{1D78D}
1311 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho}{1D78E}
1312 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi}{1D78F}
1313 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itpartial}{\um@usv@bfsfitpartial}
1314 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarepsilon}{1D7C4}
1315 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvartheta}{1D7C5}
1316 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarkappa}{1D7C6}
1317 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarphi}{1D7C7}
1318 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarrho}{1D7C8}
1319 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@itvarpi}{1D7C9}
1320 }{
1321 \um_set_mathalphabet_greek:Nnn \mathbfsf {\um@usv@upgreek,\um@usv@itgreek}{\um_bfsf_gree
1322 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@partial,\um@usv@itpartial}{\um_bfsfpart
1323 \bool_if:NTF \g_um_upsans_bool {
1324 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7
1325 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1326 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1327 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1328 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{1D78E}
1329 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1330 }{
1331 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varepsilon,\um@usv@itvarepsilon}{1D7
1332 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@vartheta,\um@usv@itvartheta}{1D7C5}
1333 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varkappa,\um@usv@itvarkappa}{1D7C6}
1334 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varphi,\um@usv@itvarphi}{1D7C7}
1335 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varrho,\um@usv@itvarrho}{1D7C8}
1336 \um_set_mathalphabet_char:Nnn \mathbfsf {\um@usv@varpi,\um@usv@itvarpi}{1D7C9}
1337 }
1338 }

```

1339 }

7.2.7 Bold upright sans serif: `\mathbfsfup`

```
1340 \cs_new:Npn \um_config_mathbfsfup_num: {
1341   \um_set_mathalphabet_numbers:Nnn{\mathbfsfup}{\um@usv@num}{\um@usv@bfsfnum}
1342 }
1343 \cs_new:Npn \um_config_mathbfsfup_Latin: {
1344   \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfu
1345 }
1346 \cs_new:Npn \um_config_mathbfsfup_latin: {
1347   \um_set_mathalphabet_latin:Nnn{\mathbfsfup}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfu
1348   \um_set_mathalphabet_char:Nnn \mathbfsfup {\um@usv@ith} {"1D5F5}
1349 }
1350 \cs_new:Npn \um_config_mathbfsfup_Greek: {
1351   \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfu
1352   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varTheta,\um@usv@itvarTheta}{1D767}
1353   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@Nabla,\um@usv@itNabla}{1D76F}
1354 }
1355 \cs_new:Npn \um_config_mathbfsfup_greek: {
1356   \um_set_mathalphabet_greek:Nnn{\mathbfsfup}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfu
1357   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@partial,\um@usv@itpartial}{1D789}
1358   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varepsilon,\um@usv@itvarepsilon}{1D78A}
1359   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@vartheta,\um@usv@itvartheta}{1D78B}
1360   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varkappa,\um@usv@itvarkappa}{1D78C}
1361   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varphi,\um@usv@itvarphi}{1D78D}
1362   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varrho,\um@usv@itvarrho}{1D78E}
1363   \um_set_mathalphabet_char:Nnn{\mathbfsfup}{\um@usv@varpi,\um@usv@itvarpi}{1D78F}
1364 }
```

7.2.8 Bold italic sans serif: `\mathbfsfit`

```
1365 \cs_new:Npn \um_config_mathbfsfit_num: {
1366   \um_set_mathalphabet_numbers:Nnn{\mathbfsfit}{\um@usv@num}{\um@usv@bfsfnum}
1367 }
1368 \cs_new:Npn \um_config_mathbfsfit_Latin: {
1369   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@upLatin,\um@usv@itLatin}{\um@usv@bfsfi
1370 }
1371 \cs_new:Npn \um_config_mathbfsfit_latin: {
1372   \um_set_mathalphabet_latin:Nnn{\mathbfsfit}{\um@usv@uplatin,\um@usv@itlatin}{\um@usv@bfsfi
1373   \um_set_mathalphabet_char:Nnn \mathbfsfit {\um@usv@ith} {"1D65D}
1374 }
1375 \cs_new:Npn \um_config_mathbfsfit_Greek: {
1376   \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upGreek,\um@usv@itGreek}{\um@usv@bfsfi
1377   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varTheta}{1D7A1}
1378   \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@Nabla,\um@usv@itNabla}{\um@usv@bfsfitNa
1379 }
1380 \cs_new:Npn \um_config_mathbfsfit_greek: {
```

```

1381 \um_set_mathalphabet_greek:Nnn{\mathbfsfit}{\um@usv@upgreek,\um@usv@itgreek}{\um@usv@bfsfi
1382 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@partial,\um@usv@itpartial}{\um@usv@bfsf
1383 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varepsilon,\um@usv@itvarepsilon}{ "1D7C4
1384 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@vartheta,\um@usv@itvartheta}{ "1D7C5}
1385 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varkappa,\um@usv@itvarkappa}{ "1D7C6}
1386 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varphi,\um@usv@itvarphi}{ "1D7C7}
1387 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varrho,\um@usv@itvarrho}{ "1D7C8}
1388 \um_set_mathalphabet_char:Nnn{\mathbfsfit}{\um@usv@varpi,\um@usv@itvarpi}{ "1D7C9}
1389 }

```

7.3 Definitions of the math symbols

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

`\um@scancharlet` We need to do some trickery to transform the `\UnicodeMathSymbol` argument
`\um@scanactivedef` "ABCDEF into the \TeX ‘caret input’ form `^^^abcdef`. It is *very important* that the argument has five characters. Otherwise we need to change the number of ^ chars.

To do this, turn ^ into a regular ‘other’ character and define the macro to perform the lowercasing and `\let`. `\scantokens` changes the carets back into their original meaning after the group has ended and ^’s catcode returns to normal.

```

1390 \begingroup
1391 \char_make_other:N \^
1392 \cs_gset:Npn \um@scancharlet#1="#2\@nil {
1393   \lowercase{
1394     \scantokens{\global\let#1=^^^#2}
1395   }
1396 }

```

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

```

1397 \gdef\um@scanactivedef"#1\@nil#2{
1398   \lowercase{
1399     \tl_rescan:nn{
1400       \ExplSyntaxOn
1401       \char_make_math_superscript:N\^
1402     }{
1403       \global\def^^^#1{#2}
1404     }
1405   }
1406 }
1407 \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`.

```

1408 \begingroup
1409   \def\UnicodeMathSymbol#1#2#3#4{
1410     \um@scancharlet#2=#1\@nil
1411   }
1412   \char_make_other:N \#
1413   \@input{unicode-math-table.tex}
1414 \endgroup
Fix \backslash:
1415 \group_begin:
1416   \lcode` \*=` \
1417   \char_make_escape:N \
1418   \char_make_other:N \
1419   |lowercase{
1420   |group_end:|let|backslash=*}

```

8 Epilogue

Lots of little things to tidy up.

8.0.1 Primes

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

```

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

```

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

```

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

```

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

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

- Prime encountered; pcount=1.

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

```

1421 \muskip_new:N \g_um_primekern_muskip
1422 \muskip_gset:Nn \g_um_primekern_muskip { -\thinmuskip/2 }% arbitrary
1423 \num_new:N \l_um_primecount_num

1424 \cs_new:Nn \um_nprimes:n {
1425   ^{
1426     \primesingle
1427     \prg_replicate:nn {#1-1} { \mskip \g_um_primekern_muskip \primesingle }
1428   }
1429 }

1430 \cs_new:Nn \um_nprimes_select:n {
1431   \prg_case_int:nnn {#1}{
1432     {1} { ^{\primesingle} }
1433     {2} {
1434       \um_glyph_if_exist:nTF {"2033} { ^{\primedouble} } {\um_nprimes:n {#1}}
1435     }
1436     {3} {
1437       \um_glyph_if_exist:nTF {"2034} { ^{\primetripel} } {\um_nprimes:n {#1}}
1438     }
1439     {4} {
1440       \um_glyph_if_exist:nTF {"2057} { ^{\primequadruple} } {\um_nprimes:n {#1}}
1441     }
1442   }{
1443     \um_nprimes:n {#1}
1444   }
1445 }

```

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.

```

1446 \cs_new:Nn \um_scanprime: {
1447   \num_zero:N \l_um_primecount_num
1448   \um_scanprime_collect:
1449 }

```

```

1450 \cs_new:Nn \um_scanprime_collect: {
1451   \num_incr:N \l_um_primecount_num
1452   \peek_meaning_remove:NTF ' {
1453     \um_scanprime_collect:
1454   }{
1455     \peek_meaning_remove:NTF \um_scanprime: {
1456       \um_scanprime_collect:
1457     }{
1458       \peek_meaning_remove:NTF ^^^^2032 {
1459         \um_scanprime_collect:
1460       }{
1461         \um_nprimes_select:n {\l_um_primecount_num}
1462       }
1463     }
1464   }
1465 }

1466 \cs_set_eq:NN \prime \um_scanprime:
1467 \group_begin:
1468   \char_make_active:N \
1469   \char_make_active:n {"2032}
1470   \cs_gset_eq:NN ' \um_scanprime:
1471   \cs_gset_eq:NN ^^^^2032 \um_scanprime:
1472 \group_end:

```

8.0.2 Unicode radicals

Undo the damage made to `\sqrt`:

```

1473 \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`.

```

1474 \def\r@@t#1#2{
1475   \setbox\z@\hbox{$\m@th #1\sqrtsign{#2}$}
1476   \um@scaled@apply{#1}{\kern}{\fontdimen63\um@font}
1477   \raise \dimexpr(
1478     \um@fontdimen@percent{65}{\um@font}\ht\z@-
1479     \um@fontdimen@percent{65}{\um@font}\dp\z@
1480   )\relax
1481   \copy \rootbox
1482   \um@scaled@apply{#1}{\kern}{\fontdimen64\um@font}
1483   \box \z@
1484 }

```

8.0.3 Unicode sub- and super-scripts

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

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

First, the setup of each mathactive char:

```
1485 \prop_new:N \g_um_supers_prop
1486 \prop_new:N \g_um_subs_prop
1487 \cs_generate_variant:Nn \prop_gput:Nnn {Nxn}
1488 \cs_generate_variant:Nn \prop_get:NnN {cxN}
1489 \cs_generate_variant:Nn \prop_if_in:NnTF {cx}
1490
1491 \group_begin:
1492
1493 % Populate a property list with superscript characters; their mean-
1494 % ing as their key,
1495 % for reasons that will become apparent soon, and their replace-
1496 % ment as each key's value.
1497 % Then make the superscript active and bind it to the scanning function.
1498 %
1499 % \cs{scantokens} makes this process much simpler since we can acti-
1500 % vate the char
1501 % and assign its meaning in one step.
1502 \cs_set:Nn \um_setup_active_superscript:nn {
1503   \prop_gput:Nxn \g_um_supers_prop {\meaning #1} {#2}
1504   \char_make_active:n `{#1}
1505   \global\XeTeXmathcodenum`#1 = "1FFFFF \scan_stop:
1506   \scantokens{
1507     \cs_gset:Npn #1 {
1508       \tl_set:Nn \l_um_ss_chain_tl {#2}
1509       \cs_set_eq:NN \um_sub_or_super:n \sp
1510       \tl_set:Nn \l_um_tmpa_tl {supers}
1511       \um_scan_sscript:
1512     }
1513   }
1514 }
1515
1516 \um_setup_active_superscript:nn {^^^2070} {0}
1517 \um_setup_active_superscript:nn {^^^00b9} {1}
1518 \um_setup_active_superscript:nn {^^^00b2} {2}
1519 \um_setup_active_superscript:nn {^^^00b3} {3}
```

```

1517 \um_setup_active_superscript:nn {^^^^2074} {4}
1518 \um_setup_active_superscript:nn {^^^^2075} {5}
1519 \um_setup_active_superscript:nn {^^^^2076} {6}
1520 \um_setup_active_superscript:nn {^^^^2077} {7}
1521 \um_setup_active_superscript:nn {^^^^2078} {8}
1522 \um_setup_active_superscript:nn {^^^^2079} {9}
1523 \um_setup_active_superscript:nn {^^^^207a} {+}
1524 \um_setup_active_superscript:nn {^^^^207b} {-}
1525 \um_setup_active_superscript:nn {^^^^207c} {=}
1526 \um_setup_active_superscript:nn {^^^^207d} {(}
1527 \um_setup_active_superscript:nn {^^^^207e} {)}
1528 \um_setup_active_superscript:nn {^^^^207i} {i}
1529 \um_setup_active_superscript:nn {^^^^207f} {n}
1530
1531 % Ditto above.
1532 \cs_set:Nn \um_setup_active_subscript:nn {
1533   \prop_gput:Nxn \g_um_subs_prop {\meaning #1} {#2}
1534   \char_make_active:n {\`#1}
1535   \global\XeTeXmathcodenum `#1 = "1FFFFF \scan_stop:
1536   \scantokens{
1537     \cs_gset:Npn #1 {
1538       \tl_set:Nn \l_um_ss_chain_tl {#2}
1539       \cs_set_eq:NN \um_sub_or_super:n \sb
1540       \tl_set:Nn \l_um_tmpa_tl {subs}
1541       \um_scan_sscript:
1542     }
1543   }
1544 }
1545
1546 \um_setup_active_subscript:nn {^^^^2080} {0}
1547 \um_setup_active_subscript:nn {^^^^2081} {1}
1548 \um_setup_active_subscript:nn {^^^^2082} {2}
1549 \um_setup_active_subscript:nn {^^^^2083} {3}
1550 \um_setup_active_subscript:nn {^^^^2084} {4}
1551 \um_setup_active_subscript:nn {^^^^2085} {5}
1552 \um_setup_active_subscript:nn {^^^^2086} {6}
1553 \um_setup_active_subscript:nn {^^^^2087} {7}
1554 \um_setup_active_subscript:nn {^^^^2088} {8}
1555 \um_setup_active_subscript:nn {^^^^2089} {9}
1556 \um_setup_active_subscript:nn {^^^^208a} {+}
1557 \um_setup_active_subscript:nn {^^^^208b} {-}
1558 \um_setup_active_subscript:nn {^^^^208c} {=}
1559 \um_setup_active_subscript:nn {^^^^208d} {(}
1560 \um_setup_active_subscript:nn {^^^^208e} {)}
1561 \um_setup_active_subscript:nn {^^^^2090} {a}
1562 \um_setup_active_subscript:nn {^^^^2091} {e}

```



```

1563 \um_setup_active_subscript:nn {^^^1d62} {i}
1564 \um_setup_active_subscript:nn {^^^2092} {o}
1565 \um_setup_active_subscript:nn {^^^1d63} {r}
1566 \um_setup_active_subscript:nn {^^^1d64} {u}
1567 \um_setup_active_subscript:nn {^^^1d65} {v}
1568 \um_setup_active_subscript:nn {^^^2093} {x}
1569 \um_setup_active_subscript:nn {^^^1d66} {\beta}
1570 \um_setup_active_subscript:nn {^^^1d67} {\gamma}
1571 \um_setup_active_subscript:nn {^^^1d68} {\rho}
1572 \um_setup_active_subscript:nn {^^^1d69} {\phi}
1573 \um_setup_active_subscript:nn {^^^1d6a} {\chi}
1574
1575 \group_end:
1576
1577 % The scanning command, evident in its purpose:
1578 \cs_new:Nn \um_scan_sscript: {
1579   \um_scan_sscript:TF {
1580     \um_scan_sscript:
1581   }{
1582     \um_sub_or_super:n {\l_um_ss_chain_tl}
1583   }
1584 }
1585
1586 % The main theme here is stolen from the source to the vari-
1587   ous \cs{peek_} functions.
1588 % Consider this function as simply boilerplate:
1589 \cs_new:Nn \um_scan_sscript:TF {
1590   \tl_set:Nx \l_peek_true_aux_tl { \exp_not:n{ #1 } }
1591   \tl_set_eq:NN \l_peek_true_tl \c_peek_true_remove_next_tl
1592   \tl_set:Nx \l_peek_false_tl { \exp_not:n{ \group_align_safe_end: #2 } }
1593   \group_align_safe_begin:
1594   \peek_after:NN \um_peek_execute_branches_ss:
1595 }
1596
1597 % We do not skip spaces when scanning ahead, and we explicitly wish to
1598 % bail out on encountering a space or a brace.
1599 \cs_new:Npn \um_peek_execute_branches_ss: {
1600   \bool_if:nTF {
1601     \token_if_eq_catcode_p:NN \l_peek_token \c_group_begin_token ||
1602     \token_if_eq_catcode_p:NN \l_peek_token \c_group_end_token ||
1603     \token_if_eq_meaning_p:NN \l_peek_token \c_space_token
1604   } { \l_peek_false_tl }
1605   { \um_peek_execute_branches_ss_aux: }
1606 }
1607

```

```

1608 % This is the actual comparison code.
1609 % Because the peeking has already tokenised the next token,
1610 % it's too late to extract its charcode directly. Instead,
1611 % we look at its meaning, which remains a `character' even
1612 % though it is itself math-active. If the character is ever
1613 % made fully active, this will break our assumptions!
1614 %
1615 % If the char's meaning exists as a property list key, we
1616 % build up a chain of sub-/superscripts and iterate. (If not, exit and
1617 % typeset what we've already collected.)
1618 \cs_new:Nn \um_peek_execute_branches_ss_aux: {
1619   \prop_if_in:cxTF
1620     {g_um_\l_um_tmpa_tl _prop}
1621     {\meaning\l_peek_token}
1622   {
1623     \prop_get:cxN
1624       {g_um_\l_um_tmpa_tl _prop}
1625       {\meaning\l_peek_token}
1626     \l_um_tmpb_tl
1627     \tl_put_right:NV \l_um_ss_chain_tl \l_um_tmpb_tl
1628     \l_peek_true_tl
1629   }
1630   {\l_peek_false_tl}
1631 }

```

8.0.4 Synonyms and all the rest

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

```

1632 \def\operator@font{\um_setup_mathup:}
1633 \def\to{\rightarrow}
1634 \def\vec{\overrightarrow}
1635 \def\le{\leq}
1636 \def\ge{\geq}
1637 \def\neq{\neq}

```

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

```

1638 \@ifpackageloaded{amsmath}{
1639   % define their own colon, perhaps I should just steal it.
1640 }{
1641   \cs_set_protected:Npn \colon {
1642     \bool_if:NTF \g_um_literal_colon_bool {:} { \mathpunct{:} }
1643   }
1644 }

```

`\mathcal`

```
1645 \def\mathcal{\mathscr}
```

`\mathrm`

```
1646 \def\mathrm{\mathup}
```

8.0.5 Compatibility

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

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

```
1647 \ifpackageloaded{amsmath}{
1648   \tl_remove_in:Nn \@begindocumenthook {
1649     \mathchardef\std@minus\mathcode`\-\relax
1650     \mathchardef\std@equal\mathcode`\=\relax
1651   }
1652 }{}
```

- 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:

```
1653 \ifpackageloaded{amsopn}{
1654   \cs_set:Npn \newmcodes@ {
1655     \mathcode`\'39
1656     \mathcode`\*42
1657     \mathcode`\."613A%
1658     % \ifnum\mathcode`\- =45 \else
1659     %   \mathchardef\std@minus\mathcode`\-\relax
1660     % \fi
1661     \mathcode`\-45
1662     \mathcode`\ /47
1663     \mathcode`\."603A\relax
1664   }
1665 }{}
```

Octothorpe is an odd one:

```
1666 \AtBeginDocument{
1667   \def\#{\mode_if_math:TF{\mathoctothorpe}{\char`\#}}
1668   \def\widehat{\hat}
1669   \def\widetilde{\tilde}
1670 }
```

Overriding amsmath definitions:

```
1671 \AtBeginDocument{
1672   \def\@cdots{\mathinner{\cdots}}
1673 }
```

Interaction with beamer:

```
1674 \@ifclassloaded{beamer}{
1675   \ifbeamer@suppressreplacements\else
1676     \PackageWarningNoLine{unicode-math}{
1677       Disabling~ beamer's~ math~ setup.^{}
1678       Please~ load~ beamer~ with~ the~ [professionalfonts]~ class~ option
1679     }
1680     \beamer@suppressreplacementstrue
1681   \fi
1682 }{}
```

The end.

```
1683 \ExplSyntaxOff
```

File II

STIX table data extraction

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

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

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

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

```

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

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

```

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

```

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

```

Print the actual entry corresponding to the unicode character:

```

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

```

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

```

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

```

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

```

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

```

Fixing up a couple of things in the STIX table.

```

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

```

A Documenting maths support in the NFSS

A.1 Overview

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

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

`\DeclareSymbolFont{<name>}\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@}}}
```

File III

X_YTeX math font dimensions

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

<code>\fontdimen</code>	Dimension name	Description
10	<code>SCRIPTPERCENTSCALEDOWN</code>	Percentage of scaling down for script level 1. Suggested value: 80%.
11	<code>SCRIPTSCRIPTPERCENTSCALEDOWN</code>	Percentage of scaling down for script level 2 (ScriptScript). Suggested value: 60%.
12	<code>DELIMITEDSUBFORMULAMINHEIGHT</code>	Minimum height required for a delimited expression to be treated as a subformula. Suggested value: normal line height \times 1.5.
13	<code>DISPLAYOPERATORMINHEIGHT</code>	Minimum height of n-ary operators (such as integral and summation) for formulas in display mode.
14	<code>MATHLEADING</code>	White space to be left between math formulas to ensure proper line spacing. For example, for applications that treat line gap as a part of line ascender, formulas with ink going above (<code>os2.sTypoAscender</code> + <code>os2.sTypoLineGap</code> – <code>MathLeading</code>) or with ink going below <code>os2.sTypoDescender</code> will result in increasing line height.
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 overshots.

\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	STACKBOTTOMDISPLAYSTYLE-SHIFTDOWN	Standard shift down applied to the bottom element of a stack in display style. Positive for moving in the downward direction.
36	STACKGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element. Suggested: 3×default rule thickness.
37	STACKDISPLAYSTYLEGAPMIN	Minimum gap between (ink) bottom of the top element of a stack, and the (ink) top of the bottom element in display style. Suggested: 7×default rule thickness.
38	STRETCHSTACKTOPSHIFTUP	Standard shift up applied to the top element of the stretch stack.
39	STRETCHSTACKBOTTOMSHIFT-DOWN	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	
\"	17
\#	1412, 1667
\'	1468, 1655
*	485, 1416, 1656
\-	484, 1649, 1658, 1659, 1661
\.	1657
\/	1662
\:	487, 1663
\::	740
\::f	740
\::n	740
\=	1650
\@DeclareMathDelimiter	331
\@DeclareMathSizes	296
\@backslashchar	930
\@beginofdocumenthook	1648
\@cclvi	356
\@cdots	1672
\@elt	605–609, 612, 616, 618
\@empty	407, 408, 445, 534, 656, 663, 678, 699, 704
\@ifclassloaded	1674
\@ifnextchar	1473
\@ifpackageloaded	1638, 1647, 1653
\@ii	662, 663, 665, 667, 670, 675
\@input	462, 1413
\@marker	675, 694
\@nil	365, 515, 675, 688–691, 1392, 1397, 1410
\@preamblecmds	340
\@sqrt	1473
\@tempa	133, 180, 212, 238, 256, 274, 282, 418, 434, 654, 666, 690, 692, 704
\@tempb	133, 134, 180, 181, 212, 213, 238, 239, 256, 257, 274, 275, 282, 283, 654, 655, 693, 694, 699
\@tempswafalse	664
\@tempswatruetrue	668, 671, 696, 701, 706, 711
\@um@bfliteraltrue	175, 200
\@um@bfupGreekfalse	139, 182
\@um@bfupGreektrue	152, 165, 188, 194
\@um@bfupLatinfalse	142, 184
\@um@bfupLatintrue	155, 168, 190, 196
\@um@bfupgreekfalse	140, 153, 183, 189
\@um@bfupgreektrue	166, 195
\@um@bfuplatinfalse	143, 185
\@um@bfuplatintrue	156, 169, 191, 197
\@um@fontspec@featuretrue	409
\@um@literaltrue	174
\@um@ot@math@true	437
\@um@upGreekfalse	135
\@um@upGreektrue	148, 161
\@um@upLatinfalse	137, 150
\@um@upLatintrue	163
\@um@upNablafalse	144, 242
\@um@upNablatruetrue	157, 170, 240
\@um@upgreekfalse	136, 149
\@um@upgreektrue	162
\@um@uplatinfalse	138, 151, 164
\@um@uppartialfalse	141, 154, 186, 192, 260
\@um@uppartialtrue	167, 198, 258
\@xDeclareMathDelimiter	332
\@xxDeclareMathDelimiter	330
\\	10–13, 17, 23–33, 1416, 1418
\^	33, 1391, 1401
\	1417
Numbers	
\0	23
_	17–20, 23–32
A	
\A	24
\a	25
\addnolimits	<u>611</u>
\addtoversion	<u>303</u>
\alloc@	356

\Alpha	767	\clist_map_inline:nn	
\alpha	792	520, 586, 593, 917, 924
\alpha@elt	307	\clist_map_variable:NNn	662
\alpha@list	306	\clist_map_variable:nNn ..	719, 742, 748
\AtBeginDocument	765, 1666, 1671	\colon	1641
\awint	607	\copy	1481
B		\cs	1497, 1586
\B	1026	\cs_generate_variant:Nn ...	1487–1489
\beamer@suppressreplacementstrue	1680	\cs_gset:cpn	370, 380
\begingroup	362, 615, 1390, 1408	\cs_gset:Npn ..	383, 388, 1392, 1504, 1537
\Beta	768	\cs_gset:Npx	393
\beta	793, 1569	\cs_gset_eq:NN	1470, 1471
\bgroup	954	\cs_if_exist:cF	949
\bool_if:NF	486	\cs_new:Nn	469,
\bool_if:NTF	222, 797, 816, 822,	483, 514, 519, 524, 527, 741, 747,	
827, 1055, 1063, 1069, 1263, 1271,		756, 759, 762, 897, 916, 948, 1424,	
1277, 1285, 1295, 1303, 1323, 1642		1430, 1446, 1450, 1578, 1588, 1618	
\bool_if:nTF	1599	\cs_new:Npn ..	963, 966, 970, 975, 985,
\bool_new:N	22, 209, 210, 281	988, 992, 997, 1007, 1011, 1021,	
\bool_set_false:N		1024, 1035, 1042, 1050, 1054,	
... 145, 146, 172, 177, 214, 276, 286		1062, 1076, 1079, 1082, 1086,	
\bool_set_true:N		1089, 1092, 1095, 1099, 1102,	
158, 159, 171, 176, 216, 218, 278, 284		1105, 1109, 1112, 1120, 1134,	
\box	1483	1153, 1192, 1195, 1198, 1202,	
C		1207, 1218, 1221, 1224, 1228,	
\C	1013, 1044	1234, 1245, 1248, 1252, 1255,	
\c_group_begin_token	1600	1259, 1262, 1270, 1284, 1302,	
\c_group_end_token	1601	1340, 1343, 1346, 1350, 1355,	
\c_peek_true_remove_next_tl ...	1590	1365, 1368, 1371, 1375, 1380, 1598	
\c_space_token	1602	\cs_set:cpn	950
\cdots	1672	\cs_set:Nn	203, 221, 245,
\cdp@elt	294	263, 402, 475, 478, 533, 585, 590,	
\cdp@list	293	718, 728, 731, 734, 737, 832, 843,	
\char	1667	853, 860, 869, 878, 942, 1499, 1532	
\char_make_active:N	1468	\cs_set:Npn	740, 1654
\char_make_active:n 363, 1469, 1501, 1534		\cs_set_eq:cN	943
\char_make_escape:N	1417	\cs_set_eq:NN	
\char_make_math_superscript:N ..	1401	448–451, 455–458, 1466, 1506, 1539	
\char_make_other:N ..	1391, 1412, 1418	\cs_set_protected:cpn	953
\chardef	356	\cs_set_protected:Npn	1641
\Chi	789	\cs_to_str:N	365, 370, 380, 600
\chi	818, 1573	\csname	359, 365,
\cirfnint	607	368, 371, 403, 410, 529, 918, 925, 957	
\clist_map_break:	921	D	
\clist_map_inline:Nn	591	\DeclareDocumentCommand	405
		\DeclareMathAccent	324

<code>\l_um_sscript_font_tl</code>	416, 427	<code>\mathcode</code>	364,
<code>\l_um_tmpa_tl</code>	1507, 1540, 1620, 1624		1649, 1650, 1655–1659, 1661–1663
<code>\l_um_tmpb_tl</code>	1626, 1627	<code>\mathfrak</code>	1043–1048, 1051, 1052
<code>\Lambda</code>	777	<code>\mathgroup</code>	356
<code>\lambda</code>	804	<code>\mathinner</code>	1672
<code>\lccode</code>	1416	<code>\mathit</code>	986, 989, 990, 993–995, 998–1005
<code>\le</code>	1635	<code>\mathoctothorpe</code>	1667
<code>\left</code>	<u>624</u>	<code>\mathop</code>	361
<code>\left@primitive</code>	624, 625	<code>\mathopen</code>	382, 625
<code>\leq</code>	1635	<code>\mathord</code>	490–493,
<code>\let</code>	357, 406–408, 624, 656, 1394		495, 496, 499–506, 508–511, 525
<code>\lowercase</code>	1393, 1398	<code>\mathpunct</code>	1642
<code>\lowint</code>	609	<code>\mathrel</code>	487
M			
<code>\M</code>	1032	<code>\mathrm</code>	<u>1646</u>
<code>\m@th</code>	1475	<code>\mathscr</code>	1025–1033, 1036–1040, 1645
<code>\mathaccent</code>	392	<code>\mathsf</code>	1056, 1057, 1059,
<code>\mathalpha</code>	601, 723		1064–1066, 1068, 1070, 1072, 1077
<code>\mathbb</code>	1008, 1009, 1012–1019, 1022	<code>\mathsf fit</code>	1090, 1093, 1096, 1097
<code>\mathbf</code>	1110, 1114,	<code>\mathsf fup</code>	1080, 1083, 1084, 1087
	1115, 1117, 1122–1124, 1126,	<code>\mathtt</code>	1100, 1103, 1106, 1107
	1128, 1130, 1135, 1137–1142,	<code>\mathup</code>	
	1144, 1145, 1147, 1149, 1154,		964, 967, 968, 971–973, 976–983, 1646
	1156–1171, 1173, 1175–1180,	<code>\mddefault</code>	461
	1182–1187, 1189, 1232, 1243	<code>\meaning</code>	1500, 1533, 1621, 1625
<code>\mathbffrac</code>	1246, 1249, 1250	<code>\mitAlpha</code>	767
<code>\mathbf fit</code>	1193, 1196,	<code>\mitalpha</code>	792
	1199, 1200, 1203–1205, 1208–1216	<code>\mitBeta</code>	768
<code>\mathbf fscr</code>	1253, 1256, 1257	<code>\mitbeta</code>	793
<code>\mathbfsf</code>	1260, 1264,	<code>\mitChi</code>	789
	1265, 1267, 1272–1274, 1276,	<code>\mitchi</code>	818
	1278, 1280, 1286–1291, 1293,	<code>\mitDelta</code>	770
	1294, 1296, 1298, 1304–1319,	<code>\mitdelta</code>	795
	1321, 1322, 1324–1329, 1331–1336	<code>\mitEpsilon</code>	771
<code>\mathbfsf fit</code>	1366, 1369,	<code>\mitepsilon</code>	797, 822
	1372, 1373, 1376–1378, 1381–1388	<code>\mitEta</code>	773
<code>\mathbfsfup</code>	1341, 1344,	<code>\miteta</code>	800
	1347, 1348, 1351–1353, 1356–1363	<code>\mitGamma</code>	769
<code>\mathbfup</code>	1219, 1222,	<code>\mitgamma</code>	794
	1225, 1226, 1229–1231, 1235–1242	<code>\mitIota</code>	775
<code>\mathbin</code>	484, 485	<code>\mitiota</code>	802
<code>\mathcal</code>	<u>1645</u>	<code>\mitKappa</code>	776
<code>\mathchar@type</code>	336, 369,	<code>\mitkappa</code>	803
	383, 385, 388, 390, 393, 395, 403, 528	<code>\mitLambda</code>	777
<code>\mathchardef</code>	1649, 1650, 1659	<code>\mitlambda</code>	804
<code>\mathclose</code>	387	<code>\mitMu</code>	778
		<code>\mitmu</code>	805

\backslash Phi	788	\backslash restore@mathversion	308
\backslash phi	815, 1572	\backslash Rho	783
\backslash Pi	782	\backslash rho	810, 1571
\backslash pi	809	\backslash rightarrow	1633
\backslash pointint	608	\backslash rootbox	1481
\backslash prg_case_int:nnn	1431	\backslash rppolint	607
\backslash prg_do_nothing:	943		
\backslash prg_new_conditional:Nnn	945	S	
\backslash prg_replicate:nn	1427	\backslash sb	1539
\backslash prg_return_false:	946	\backslash scan_stop: ...	530, 531, 946, 1502, 1535
\backslash prg_return_true:	946	\backslash scantokens	1394, 1503, 1536
\backslash prg_stepwise_variable:nnnNn	720, 749	\backslash scpolint	608
\backslash prime	1466	\backslash scriptscriptstyle	349
\backslash primedouble	1434	\backslash scriptstyle	346
\backslash primequadruple	1440	\backslash set@@mathdelimiter	334
\backslash primesingle	525, 1426, 1427, 1432	\backslash set@mathaccent	325
\backslash primetripel	1437	\backslash set@mathchar	327
\backslash process@table	311	\backslash set@mathdelimiter	333
\backslash ProcessOptionsX	290	\backslash set@mathsymbol	328
\backslash prop_get:cxN	1623	\backslash setbox	1475
\backslash prop_get:NnN	1488	\backslash setkeys	417
\backslash prop_gput:Nnn	1487	\backslash SetMathAlphabet	322
\backslash prop_gput:Nxn	1500, 1533	\backslash SetMathAlphabet@	323
\backslash prop_if_in:cxTF	1619	\backslash setmathfont	405, 640
\backslash prop_if_in:NnTF	1489	\backslash SetSymbolFont	317
\backslash prop_new:N	1485, 1486	\backslash SetSymbolFont@	318
\backslash protect	640	\backslash sf@size	422, 426
\backslash ProvidesPackage	1	\backslash Sigma	785
\backslash Psi	790	\backslash sigma	812
\backslash psi	819	\backslash sp	1506
		\backslash space	928, 957
Q		\backslash sqint	608
\backslash Q	1017	\backslash sqrt	623, 1473
		\backslash sqrtsign	1473, 1475
R		\backslash std@equal	1650
\backslash R	1018, 1033, 1047	\backslash std@minus	1649, 1659
\backslash r@@t	1474	\backslash stepcounter	453
\backslash raise	1477	\backslash string	368, 371, 689, 690
\backslash relax	134, 181,	\backslash strip@pt	343
	213, 239, 257, 275, 283, 343, 361,	\backslash sumint	606
	364, 369, 380, 382–385, 387–390,		
	392, 393, 395, 403, 406, 435, 436,	T	
	592, 655, 667, 670, 694, 695, 700,	\backslash Tau	786
	705, 709, 710, 722, 724, 744, 751,	\backslash tau	813
	752, 1480, 1649, 1650, 1659, 1663	\backslash tf@size	421, 422
\backslash removenolimits	614	\backslash Theta	774
\backslash RequirePackage	3–5	\backslash theta	801

<code>\theum@fam</code>	454	<code>\um@usv@bffraklatin</code>	67, 1249
<code>\thinmuskip</code>	1422	<code>\um@usv@bfGreek</code>	60
<code>\tilde</code>	1669	<code>\um@usv@bfgreek</code>	61
<code>\tl_map_inline:nn</code>	291	<code>\um@usv@bfitGreek</code>	
<code>\tl_put_right:cn</code>	600	.. 64, 204, 558, 561, 850, 1138, 1203	
<code>\tl_put_right:Nv</code>	1627	<code>\um@usv@bfitgreek</code>	
<code>\tl_remove_in:Nn</code>	340, 1648	.. 65, 205, 565, 573, 851, 1157, 1209	
<code>\tl_rescan:nn</code>	1399	<code>\um@usv@bfith</code>	112, 1124, 1130
<code>\tl_set:Nn</code>	223–228, 230–235, 247–249, 251–253, 265–267, 269–271, 413–416, 446, 937–941, 1505, 1507, 1538, 1540	<code>\um@usv@bfitLatin</code>	62, 206, 548, 550, 846, 1115, 1196, 1208
<code>\tl_set:Nx</code>	204–207, 454, 1589, 1591	<code>\um@usv@bfitlatin</code> 63, 207, 553, 555, 847, 1123, 1199
<code>\tl_set_eq:NN</code>	1590	<code>\um@usv@bfitNabla</code>	
<code>\tl_use:c</code>	931 123, 252, 500, 508, 1142, 1205	
<code>\to</code>	1633	<code>\um@usv@bfitnum</code>	53
<code>\token_if_eq_catcode_p:NN</code> .	1600, 1601	<code>\um@usv@bfitpartial</code>	
<code>\token_if_eq_meaning_p:NN</code>	1602 129, 270, 504, 510, 1165, 1210	
U			
<code>\um@addto@mathmap</code>	587, 594, <u>599</u>	<code>\um@usv@bfitvarepsilon</code>	
<code>\um@backslash</code>	666, 689 114, 566, 574, 1166, 1182, 1211	
<code>\um@char@num@range</code> 408, 591, 678, 679, 681		<code>\um@usv@bfitvarkappa</code>	
<code>\um@char@range</code> 407, 445, 534, 656, 659, 662	 116, 568, 576, 1168, 1184, 1213	
<code>\um@firstchar</code>	665, 690	<code>\um@usv@bfitvarphi</code>	
<code>\um@firstof</code>	688–690 117, 569, 577, 1169, 1185, 1214	
<code>\um@font</code>	347, 350, 435, 436, 933, 946, 1476, 1478, 1479, 1482	<code>\um@usv@bfitvarpi</code>	
<code>\um@fontdimen@percent</code> 119, 571, 579, 1171, 1187, 1216	
..... <u>342</u> , 347, 350, 1478, 1479		<code>\um@usv@bfitvarrho</code>	
<code>\um@mathsymbol</code>	<u>358</u> , 476 118, 570, 578, 1170, 1186, 1215	
<code>\um@mversion</code>	411, 412	<code>\um@usv@bfitvarTheta</code>	
<code>\um@nolimits</code>	372, <u>604</u> , 612, 620 113, 559, 562, 1140, 1149, 1204	
<code>\um@parse@range</code>	675, <u>691</u>	<code>\um@usv@bfitvartheta</code>	
<code>\um@parse@term</code>	479, 515, <u>661</u> 115, 567, 575, 1167, 1183, 1212	
<code>\um@radicals</code>	378, <u>623</u>	<code>\um@usv@bfLatin</code>	55
<code>\um@resolve@greek</code>	<u>765</u>	<code>\um@usv@bflat</code>	57
<code>\um@scaled@apply</code>	<u>345</u> , 1476, 1482	<code>\um@usv@bfNabla</code>	
<code>\um@scanactivedef</code>	365, <u>1390</u> 122, 248, 499, 508, 1141, 1231	
<code>\um@scancharlet</code>	<u>1390</u> , 1410	<code>\um@usv@bfnum</code>	51, 1110, 1193, 1219
<code>\um@set@mathsymbol</code>	<u>359</u> , <u>360</u>	<code>\um@usv@bfpartial</code>	
<code>\um@usv@bblatin</code>	33, 1012 128, 266, 503, 510, 1158, 1236	
<code>\um@usv@bblatin</code>	34, 1008	<code>\um@usv@bfscrLatin</code>	68, 1253
<code>\um@usv@bbnum</code>	32, 1022	<code>\um@usv@bfscrLatin</code>	69, 1256
<code>\um@usv@bfDigamma</code>	95, 1135, 1232	<code>\um@usv@bfsfGreek</code>	79
<code>\um@usv@bfDigamma</code>	102, 1154, 1243	<code>\um@usv@bfsfgreek</code>	80
<code>\um@usv@bffrakLatin</code>	66, 1246	<code>\um@usv@bfsfitGreek</code> 83, 234, 1287, 1376	
		<code>\um@usv@bfsfitgreek</code> 84, 235, 1305, 1381	
		<code>\um@usv@bfsfitLatin</code> 81, 232, 1265, 1369	
		<code>\um@usv@bfsfitlatin</code> 82, 233, 1273, 1372	

$\backslash\mathrm{um@usv@bfsfitNabla}$	1203, 1229, 1287, 1293, 1351, 1376
..... 125, 253, 502, 509, 1291, 1378	
$\backslash\mathrm{um@usv@bfsfitnum}$	$\backslash\mathrm{um@usv@itgreek}$
72	31, 880, 888, 976, 998, 1157, 1173,
$\backslash\mathrm{um@usv@bfsfitpartial}$	1209, 1235, 1305, 1321, 1356, 1381
..... 131, 271, 506, 511, 1313, 1382	$\backslash\mathrm{um@usv@ith}$. 103, 836, 863, 866, 968,
$\backslash\mathrm{um@usv@bfsfLatin}$	990, 1009, 1040, 1052, 1066, 1070,
74	1072, 1084, 1097, 1107, 1124,
$\backslash\mathrm{um@usv@bfsfLatin}$	1128, 1130, 1200, 1226, 1250,
76	1257, 1274, 1278, 1280, 1348, 1373
$\backslash\mathrm{um@usv@bfsfNabla}$ 124, 249, 501, 509, 1290	$\backslash\mathrm{um@usv@itLatin}$.. 28, 834, 855, 857,
$\backslash\mathrm{um@usv@bfsfnum}$... 70, 1260, 1341, 1366	964, 986, 1012, 1025, 1043, 1057,
$\backslash\mathrm{um@usv@bfsfpartial}$	1059, 1087, 1093, 1103, 1115,
..... 130, 267, 505, 511, 1306	1117, 1196, 1208, 1222, 1246,
$\backslash\mathrm{um@usv@bfsfupGreek}$ 77, 227, 1286, 1351	1253, 1265, 1267, 1276, 1344, 1369
$\backslash\mathrm{um@usv@bfsfupgreek}$ 78, 228, 1304, 1356	$\backslash\mathrm{um@usv@itLatin}$
$\backslash\mathrm{um@usv@bfsfupLatin}$ 73, 225, 1264, 1344	29, 835, 862, 865, 967, 989, 1008,
$\backslash\mathrm{um@usv@bfsfupLatin}$ 75, 226, 1272, 1347	1036, 1051, 1065, 1068, 1083,
$\backslash\mathrm{um@usv@bfsfupnum}$	1096, 1106, 1123, 1126, 1199,
71	1225, 1249, 1256, 1273, 1347, 1372
$\backslash\mathrm{um@usv@bfupGreek}$	$\backslash\mathrm{um@usv@itNabla}$
.. 58, 204, 558, 561, 848, 1137, 1229	121, 251,
$\backslash\mathrm{um@usv@bfupgreek}$	491, 495, 972, 995, 1142, 1145,
.. 59, 205, 565, 573, 849, 1156, 1235	1205, 1231, 1291, 1294, 1353, 1378
$\backslash\mathrm{um@usv@bfuph}$	$\backslash\mathrm{um@usv@itpartial}$
111, 1128	127, 269,
$\backslash\mathrm{um@usv@bfupLatin}$	493, 496, 977, 999, 1165, 1189,
.. 54, 206, 548, 550, 844, 1114, 1222	1210, 1236, 1313, 1322, 1357, 1382
$\backslash\mathrm{um@usv@bfupLatin}$	$\backslash\mathrm{um@usv@itvarepsilon}$. 105, 881, 889,
.. 56, 207, 553, 555, 845, 1122, 1225	978, 1000, 1166, 1175, 1182, 1211,
$\backslash\mathrm{um@usv@bfupnum}$	1237, 1314, 1324, 1331, 1358, 1383
52	$\backslash\mathrm{um@usv@itvarkappa}$.. 107, 883, 891,
$\backslash\mathrm{um@usv@bfvarepsilon}$	980, 1002, 1168, 1177, 1184, 1213,
..... 96, 566, 574, 1159, 1175, 1237	1239, 1316, 1326, 1333, 1360, 1385
$\backslash\mathrm{um@usv@bfvarkappa}$	$\backslash\mathrm{um@usv@itvarphi}$ 108, 884, 892,
..... 98, 568, 576, 1161, 1177, 1239	981, 1003, 1169, 1178, 1185, 1214,
$\backslash\mathrm{um@usv@bfvarphi}$	1240, 1317, 1327, 1334, 1361, 1386
..... 99, 569, 577, 1162, 1178, 1240	$\backslash\mathrm{um@usv@itvarpi}$ 110, 886, 894,
$\backslash\mathrm{um@usv@bfvarpi}$	983, 1005, 1171, 1180, 1187, 1216,
..... 101, 571, 579, 1164, 1180, 1242	1242, 1319, 1329, 1336, 1363, 1388
$\backslash\mathrm{um@usv@bfvarrho}$	$\backslash\mathrm{um@usv@itvarrho}$ 109, 885, 893,
..... 100, 570, 578, 1163, 1179, 1241	982, 1004, 1170, 1179, 1186, 1215,
$\backslash\mathrm{um@usv@bfvarTheta}$	1241, 1318, 1328, 1335, 1362, 1387
..... 94, 559, 562, 1139, 1147, 1230	$\backslash\mathrm{um@usv@itvarTheta}$
$\backslash\mathrm{um@usv@bfvartheta}$ 104, 875, 973, 994, 1140, 1147,
..... 97, 567, 575, 1160, 1176, 1238	1149, 1204, 1230, 1296, 1298, 1352
$\backslash\mathrm{um@usv@Digamma}$	$\backslash\mathrm{um@usv@itvartheta}$.. 106, 882, 890,
86, 1135, 1232	979, 1001, 1167, 1176, 1183, 1212,
$\backslash\mathrm{um@usv@digamma}$	1238, 1315, 1325, 1332, 1359, 1384
93, 1154, 1243	
$\backslash\mathrm{um@usv@frakLatin}$	
37, 1043	
$\backslash\mathrm{um@usv@frakLatin}$	
38, 1051	
$\backslash\mathrm{um@usv@itGreek}$	
30, 840,	
871, 874, 971, 993, 1138, 1144,	

$\backslash\mathrm{um@usv@Nabla}$	120, 247, 490, 495, 972, 995, 1141, 1145, 1205, 1231, 1290, 1294, 1353, 1378
$\backslash\mathrm{um@usv@num}$	23, 535, 1022, 1077, 1080, 1090, 1100, 1110, 1193, 1219, 1260, 1341, 1366
$\backslash\mathrm{um@usv@partial}$	126, 265, 492, 496, 977, 999, 1158, 1189, 1210, 1236, 1306, 1322, 1357, 1382
$\backslash\mathrm{um@usv@scrLatin}$	35, 1025
$\backslash\mathrm{um@usv@scrLatin}$	36, 1036
$\backslash\mathrm{um@usv@sfitLatin}$	46, 230, 1057, 1093
$\backslash\mathrm{um@usv@sfitLatin}$	47, 231, 1065, 1096
$\backslash\mathrm{um@usv@sfitnum}$	41
$\backslash\mathrm{um@usv@sflatin}$	43, 223
$\backslash\mathrm{um@usv@sflatin}$	45, 224
$\backslash\mathrm{um@usv@sfnun}$	39, 1077, 1080, 1090
$\backslash\mathrm{um@usv@sfnun}$	42, 1056, 1087
$\backslash\mathrm{um@usv@sfnun}$	44, 1064, 1083
$\backslash\mathrm{um@usv@sfnun}$	40
$\backslash\mathrm{um@usv@ttLatin}$	49, 1103
$\backslash\mathrm{um@usv@ttLatin}$	50, 1106
$\backslash\mathrm{um@usv@ttnum}$	48, 1100
$\backslash\mathrm{um@usv@upGreek}$	26, 838, 871, 874, 971, 993, 1137, 1144, 1203, 1229, 1286, 1293, 1351, 1376
$\backslash\mathrm{um@usv@upgreek}$	27, 841, 880, 888, 976, 998, 1156, 1173, 1209, 1235, 1304, 1321, 1356, 1381
$\backslash\mathrm{um@usv@upLatin}$	24, 833, 855, 857, 964, 986, 1012, 1025, 1043, 1056, 1059, 1087, 1093, 1103, 1114, 1117, 1196, 1208, 1222, 1246, 1253, 1264, 1267, 1276, 1344, 1369
$\backslash\mathrm{um@usv@uplatin}$	25, 837, 862, 865, 967, 989, 1008, 1036, 1051, 1064, 1068, 1083, 1096, 1106, 1122, 1126, 1199, 1225, 1249, 1256, 1272, 1347, 1372
$\backslash\mathrm{um@usv@varepsilon}$	87, 881, 889, 978, 1000, 1159, 1175, 1182, 1211, 1237, 1307, 1324, 1331, 1358, 1383
$\backslash\mathrm{um@usv@varkappa}$	89, 883, 891, 980, 1002, 1161, 1177, 1184, 1213, 1239, 1309, 1326, 1333, 1360, 1385
$\backslash\mathrm{um@usv@varphi}$	90, 884, 892, 981, 1003, 1162, 1178, 1185, 1214, 1240, 1310, 1327, 1334, 1361, 1386
$\backslash\mathrm{um@usv@varpi}$	92, 886, 894, 983, 1005, 1164, 1180, 1187, 1216, 1242, 1312, 1329, 1336, 1363, 1388
$\backslash\mathrm{um@usv@varrho}$	91, 885, 893, 982, 1004, 1163, 1179, 1186, 1215, 1241, 1311, 1328, 1335, 1362, 1387
$\backslash\mathrm{um@usv@varTheta}$	85, 839, 872, 875, 973, 994, 1139, 1147, 1149, 1204, 1230, 1288, 1289, 1296, 1298, 1352, 1377
$\backslash\mathrm{um@usv@vartheta}$	88, 882, 890, 979, 1001, 1160, 1176, 1183, 1212, 1238, 1308, 1325, 1332, 1359, 1384
$\backslash\mathrm{um@zf@feature}$	632, 644, 647
$\backslash\mathrm{um_bf_Greek_up_or_it_usv}$	204, 1144
$\backslash\mathrm{um_bf_greek_up_or_it_usv}$	205, 1173
$\backslash\mathrm{um_bf_Latin_up_or_it_usv}$	206, 1117
$\backslash\mathrm{um_bf_latin_up_or_it_usv}$	207, 1126
$\backslash\mathrm{um_bfNabla_up_or_it_usv}$	248, 252, 508, 1145
$\backslash\mathrm{um_bfpartial_up_or_it_usv}$	266, 270, 510, 1189
$\backslash\mathrm{um_bfsf_Greek_up_or_it_usv}$	227, 234, 1293
$\backslash\mathrm{um_bfsf_greek_up_or_it_usv}$	228, 235, 1321
$\backslash\mathrm{um_bfsf_Latin_up_or_it_usv}$	225, 232, 1267
$\backslash\mathrm{um_bfsf_latin_up_or_it_usv}$	226, 233, 1276
$\backslash\mathrm{um_bfsfNabla_up_or_it_usv}$	249, 253, 509, 1294
$\backslash\mathrm{um_bfsfpartial_up_or_it_usv}$	267, 271, 511, 1322
$\backslash\mathrm{um_config_mathbb_Latin}$	1011
$\backslash\mathrm{um_config_mathbb_latin}$	1007
$\backslash\mathrm{um_config_mathbb_num}$	1021
$\backslash\mathrm{um_config_mathbf_Greek}$	1134
$\backslash\mathrm{um_config_mathbf_greek}$	1153
$\backslash\mathrm{um_config_mathbf_Latin}$	1112
$\backslash\mathrm{um_config_mathbf_latin}$	1120
$\backslash\mathrm{um_config_mathbf_num}$	1109
$\backslash\mathrm{um_config_mathbffrac_Latin}$	1245

\um_config_mathbffrak_Latin: ...	1248	\um_config_mathtt_Latin:	1105
\um_config_mathbfit_Greek:	1202	\um_config_mathtt_num:	1099
\um_config_mathbfit_greek:	1207	\um_config_mathup_Greek:	970
\um_config_mathbfit_Latin:	1195	\um_config_mathup_greek:	975
\um_config_mathbfit_latin:	1198	\um_config_mathup_Latin:	963
\um_config_mathbfit_num:	1192	\um_config_mathup_latin:	966
\um_config_mathbfscr_Latin: ...	1252	\um_glyph_if_exist:n	945
\um_config_mathbfscr_latin: ...	1255	\um_glyph_if_exist:nT	918
\um_config_mathbfsf_Greek:	1284	\um_glyph_if_exist:nTF	
\um_config_mathbfsf_greek:	1302 925, <u>945</u> , 1434, 1437, 1440	
\um_config_mathbfsf_Latin:	1262	\um_init_alphabet:n	451, 942
\um_config_mathbfsf_latin:	1270	\um_make_mathactive:nNN	525, <u>527</u>
\um_config_mathbfsf_num:	1259	\um_map_char:nn	559, 562,
\um_config_mathbfsf_Greek: ...	1375	566–571, 574–579, 737, 836, 839,	
\um_config_mathbfsf_Greek: ...	1380	863, 866, 872, 875, 881–886, 889–894	
\um_config_mathbfsf_Latin: ...	1368	\um_map_char:nn□	<u>718</u>
\um_config_mathbfsf_latin: ...	1371	\um_map_chars_greek:nn	
\um_config_mathbfsf_num:	1365	... 558, 561, 565, 573, 731, 838,	
\um_config_mathbfsfup_Greek: ...	1350	840, 841, 848–851, 871, 874, 880, 888	
\um_config_mathbfsfup_greek: ...	1355	\um_map_chars_Latin:nn	
\um_config_mathbfsfup_Latin: ...	1343	548, 550, 553, 555, 728, 833–835,	
\um_config_mathbfsfup_latin: ...	1346	837, 844–847, 855, 857, 862, 865	
\um_config_mathbfsfup_num:	1340	\um_map_chars_numbers:nn ...	535, 734
\um_config_mathbfup_Greek:	1228	\um_map_chars_range:nnn	
\um_config_mathbfup_greek:	1234 718, 729, 732, 735, 738	
\um_config_mathbfup_Latin:	1221	\um_mathmap:Nnn	449, 456, 743, 750
\um_config_mathbfup_latin:	1224	\um_mathmap_noparse:Nnn	449, <u>585</u>
\um_config_mathbfup_num:	1218	\um_mathmap_parse:Nnn	456, <u>590</u>
\um_config_mathfrak_Latin:	1042	\um_maybe_init_alphabet:n	451, 458, 919
\um_config_mathfrak_Latin:	1050	\um_Nabla_up_or_it_usv ..	247, 251, 495
\um_config_mathit_Greek:	992	\um_nprimes:n	1424, 1434, 1437, 1440, 1443
\um_config_mathit_greek:	997	\um_nprimes_select:n	1430, 1461
\um_config_mathit_Latin:	985	\um_partial_up_or_it_usv	265, 269, 496
\um_config_mathit_latin:	988	\um_peek_execute_branches_ss: ...	
\um_config_mathscr_Latin:	1024 1593, 1598	
\um_config_mathscr_latin:	1035	\um_peek_execute_branches_ss_aux:	
\um_config_mathsf_Latin:	1054 1605, 1618	
\um_config_mathsf_latin:	1062	\um_prepare_alph:n	920, <u>948</u>
\um_config_mathsf_num:	1076	\um_process_symbol_noparse:nnnn .	
\um_config_mathsf_Latin:	1092 448, <u>475</u>	
\um_config_mathsf_latin:	1095	\um_process_symbol_parse:nnnn	455, <u>475</u>
\um_config_mathsf_num:	1089	\um_remap_symbol:nnn	
\um_config_mathsfup_Latin:	1086 450, 457, 484, 485, 487,	
\um_config_mathsfup_latin:	1082	490–493, 495, 496, 499–506, 508–511	
\um_config_mathsfup_num:	1079	\um_remap_symbol_noparse:nnn	450, <u>483</u>
\um_config_mathtt_Latin:	1102		

<code>\um_remap_symbol_parse:nnn</code>	1256, 1264, 1265, 1267, 1272,
..... 457, <u>483</u> , 514	1273, 1276, 1344, 1347, 1369, 1372
<code>\um_remap_symbols:</code>	<code>\um_set_mathalphabet_numbers:Nnn</code>
..... 464, <u>483</u>	756, 1022, 1077, 1080, 1090, 1100,
<code>\um_scan_sscript:</code> 1508, 1541, 1578, 1580	1110, 1193, 1219, 1260, 1341, 1366
<code>\um_scan_sscript:TF</code>	<code>\um_set_mathcode:nnnn</code> <u>402</u> , 521, 601, 721
..... 1579, 1588	<code>\um_setup_active_subscript:nn</code> ...
<code>\um_scanprime:</code> 1532, 1546–1573
..... 1446, 1455, 1466, 1470, 1471	<code>\um_setup_active_superscript:nn</code> .
<code>\um_scanprime_collect:</code> 1499, 1513–1529
..... 1448, 1450, 1453, 1456, 1459	<code>\um_setup_alphabets:</code>
<code>\um_set_mathalph_range:Nnn</code> 467, 897
<u>747</u>	<code>\um_setup_alphanum:</code>
<code>\um_set_mathalph_range:nNnn</code> 466, <u>533</u>
..... 747, 757, 760, 763	<code>\um_setup_bf_literals:</code>
<code>\um_set_mathalphabet_char:Nnn</code> 545, <u>843</u>
..... 741, 968, 972, 973,	<code>\um_setup_bfshapes:</code>
977–983, 990, 994, 995, 999–1005, 203, <u>473</u>
1009, 1013–1019, 1026–1033,	<code>\um_setup_Greek:</code>
1037–1040, 1044–1048, 1052, 541, <u>869</u>
1066, 1070, 1072, 1084,	<code>\um_setup_greek:</code>
1097, 1107, 1124, 1128, 542, <u>878</u>
1130, 1135, 1139–1142, 1145,	<code>\um_setup_Latin:</code>
1147, 1149, 1154, 1158–1171, 539, <u>853</u>
1175–1180, 1182–1187, 1189,	<code>\um_setup_latin:</code>
1200, 1204, 1205, 1210–1216, 540, <u>860</u>
1226, 1230–1232, 1236–1243,	<code>\um_setup_literals:</code>
1250, 1257, 1274, 1278, 537, <u>832</u>
1280, 1288–1291, 1294, 1296,	<code>\um_setup_math_alphabet:nn</code>
1298, 1306–1319, 1322, 898–914, <u>916</u>
1324–1329, 1331–1336, 1348,	<code>\um_setup_mathactives:</code>
1352, 1353, 1357–1363, 465, <u>524</u>
1373, 1377, 1378, 1382–1388	<code>\um_setup_mathup:</code>
<code>\um_set_mathalphabet_char:Nnnn</code> 1632
<u>740</u>	<code>\um_setup_nabla:</code>
<code>\um_set_mathalphabet_greek:Nnn</code> 245, <u>470</u>
..... 762, 971, 976,	<code>\um_setup_partial:</code>
993, 998, 1137, 1138, 1144, 1156, 263, <u>471</u>
1157, 1173, 1203, 1209, 1229,	<code>\um_setup_sfshapes:</code>
1235, 1286, 1287, 1293, 1304, 221, <u>472</u>
1305, 1321, 1351, 1356, 1376, 1381	<code>\um_setup_shapes:</code>
<code>\um_set_mathalphabet_latin:Nnn</code> 463, <u>469</u>
..... 759,	<code>\um_sf_Latin_up_or_it_usv</code>
964, 967, 986, 989, 1008, 1012, 223, 230, 1059
1025, 1036, 1043, 1051, 1056,	<code>\um_sf_latin_up_or_it_usv</code>
1057, 1059, 1064, 1065, 1068, 224, 231, 1068
1083, 1087, 1093, 1096, 1103,	<code>\um_sub_or_super:n</code> ..
1106, 1114, 1115, 1117, 1122, 1506, 1539, 1582
1123, 1126, 1196, 1199, 1208,	<code>\um_symfont_tl</code>
1222, 1225, 1246, 1249, 1253, 446,
	454, 460, 476, 521, 529, 587, 594, 723
	<code>\UnicodeMathSymbol</code> 448, 455, 1409
	<code>\unless</code>
 663
	<code>\updefault</code>
 461
	<code>\upint</code>
 609
	<code>\Upsilon</code>
 787
	<code>\upsilon</code>
 814
	<code>\use:c</code>
 926, 951, 959
	<code>\use_none:n</code>
 458
	V
	<code>\varepsilon</code>
 821
	<code>\varkappa</code>
 825
	<code>\varointclockwise</code>
 606

<code>\varphi</code>	826	<code>\XeTeXmathaccent</code>	393
<code>\varpi</code>	830	<code>\XeTeXmathchardef</code>	367, 528
<code>\varrho</code>	829	<code>\XeTeXmathcode</code>	385, 390, 395, 403
<code>\varsigma</code>	811	<code>\XeTeXmathcodenum</code>	531, 1502, 1535
<code>\varTheta</code>	784	<code>\XeTeXradical</code>	380
<code>\vartheta</code>	824	<code>\xi</code>	780
<code>\vec</code>	1634	<code>\xi</code>	807
<code>\version@elt</code>	305	<code>\XKV@rm</code>	431
<code>\version@list</code>	304		
W		Z	
<code>\widehat</code>	1668	<code>\Z</code>	1019, 1048
<code>\widetilde</code>	1669	<code>\z@</code>	1475, 1478, 1479, 1483
X		<code>\Zeta</code>	772
<code>\xdef</code>	620, 659	<code>\zeta</code>	799
<code>\XeTeXdelcode</code>	384, 389, 626–631	<code>\zf@family</code>	461
<code>\XeTeXdelimiter</code>	383, 388	<code>\zf@fontspec</code>	418
		<code>\zf@update@ff</code>	645, 648