

Experimental Unicode mathematical typesetting: The unicode-math package

WILL ROBERTSON

Philipp Stephani, Joseph Wright, Khaled Hosny, and others

<http://github.com/wspr/unicode-math>

2023/08/13 0.8r

Contents

I	unicode-math.dtx	6
1	Package metadata	6
2	The unicode-math.sty loading file	7
II	um-code-opening.dtx	8
3	Start of the package code	8
3.1	expl3 variants	8
3.2	Low level commands	8
3.3	Primitive font commands	8
3.3.1	Mathcode and friends	9
3.3.2	NFSS-related interfaces	10
3.3.3	Font parameters	10
3.4	Alphabet Unicode positions (USVs)	11
3.5	Overcoming \@onlypreamble	11
3.6	Wrappers for kernel commands	12
III	um-code-variables.dtx	14
4	Variable initialisation	14
4.1	bool	14
4.2	int	15

4.3	tl	15
4.4	clist	16
4.5	seq	16
4.6	prop	17
4.7	muskip	17
4.8	fp	17
4.9	quark	17
IV	um-code-api.dtx	18
5	Programmers' interface	18
V	um-code-ui.dtx	19
6	The user interface commands	19
VI	um-code-pkgopt.dtx	21
7	setup and package options	21
7.1	Defaults	26
VII	um-code-msg.dtx	27
8	Error messages	27
VIII	um-code-usv.dtx	30
9	Alphabet Unicode positions	30
9.1	STIX fonts	36
IX	um-code-setchar.dtx	41
10	Setting up maths chars	41
10.1	A token list to contain the data of the math table	41
10.2	Definitions of the active math characters	41
10.3	Commands for each symbol/glyph/char	42
X	um-code-mathtext.dtx	46

11 Maths text commands	46
11.1 <code>\setmathfontface</code>	46
11.2 Hooks into $\LaTeX 2_{\epsilon}$	46
11.3 Hooks into <code>fontspec</code>	48
11.3.1 Text font	48
11.3.2 Maths font	49
XI <code>um-code-main.dtx</code>	51
12 The main <code>\setmathfont</code> macro	51
12.1 Functions for setting up symbols with <code>mathcodes</code>	57
12.2 Active math characters	57
12.3 Delimiter codes	58
12.4 (Big) operators	59
12.5 Radicals	60
12.6 <code>Fontdimens</code>	60
XII <code>um-code-fontopt.dtx</code>	62
13 Font loading options	62
13.1 Math version	62
13.2 Script and <code>scriptscript</code> font options	62
13.3 Range processing	62
XIII <code>um-code-fontparam.dtx</code>	68
14 Cross-platform interface for font parameters	68
14.1 Historical commands	73
XIV <code>um-code-mathmap.dtx</code>	74
15 Defining the math alphabets per style	74
15.1 Mapping ‘naked’ math characters	78
15.1.1 Functions	78
15.1.2 Functions for ‘normal’ alphabet symbols	79
15.2 Mapping chars inside a math style	81
15.2.1 Functions for setting up the maths alphabets	81
15.2.2 Individual mapping functions for different alphabets	82
XV <code>um-code-sym-commands.dtx</code>	84

16 Mapping in maths alphabets	84
16.1 Setting styles	84
16.2 Defining the math style macros	84
16.3 Definition of alphabets and styles	85
16.3.1 Define symbol style commands	86
16.3.2 New names for legacy textmath alphabet selection	86
16.3.3 Replacing legacy pure-maths alphabets	86
16.3.4 New commands for ambiguous alphabets	87
XVI um-code-alphabets.dtx	88
17 Setting up alphabets	88
17.1 Upright: up	88
17.2 Italic: it	89
17.3 Blackboard or double-struck: bb and bbit	91
17.4 Script and calligraphic: scr and cal	92
17.5 Fraktur or fraktur or blackletter: frak	93
17.6 Sans serif upright: sfup	93
17.7 Sans serif italic: sfit	94
17.8 Typewriter or monospaced: tt	95
17.9 Bold Italic: bfit	95
17.10 Bold Upright: bfup	97
17.11 Bold fraktur or fraktur or blackletter: bffrak	100
17.12 Bold script or calligraphic: bfscr	100
17.13 Bold upright sans serif: bsfup	101
17.14 Bold italic sans serif: bsfrit	103
XVII um-code-primes.dtx	106
18 Primes	106
XVIII um-code-sscript.dtx	113
19 Unicode sub- and super-scripts	113
XIX um-code-compat.dtx	117
20 Compatibility	117
21 Patching/augmenting 3rd-party packages	117
21.1 url	117
21.2 mathtools	117

XX	um-code-amsmath.dtx	120
22	Compatibility with amsmath	120
XXI	um-code-epilogue.dtx	123
23	Epilogue	123
23.1	Resolving Greek symbol name control sequences	123
23.1.1	Active fractions	123
23.2	Synonyms and all the rest	124
23.2.1	\not	125
23.2.2	Full-width remapping	127
23.3	Legacy characters	127

File I

unicode-math.dtx

1 *Package metadata*

List all dtx files for (a) the ins file and (b) typesetting the code.

```
1 <*dtx>
2 \def\DTXFILES{
3   \DTX{unicode-math.dtx}
4   \DTX{um-code-opening.dtx}
5   \DTX{um-code-variables.dtx}
6   \DTX{um-code-api.dtx}
7   \DTX{um-code-ui.dtx}
8   \DTX{um-code-pkgopt.dtx}
9   \DTX{um-code-msg.dtx}
10  \DTX{um-code-usv.dtx}
11  \DTX{um-code-setchar.dtx}
12  \DTX{um-code-mathtext.dtx}
13  \DTX{um-code-main.dtx}
14  \DTX{um-code-fontopt.dtx}
15  \DTX{um-code-fontparam.dtx}
16  \DTX{um-code-mathmap.dtx}
17  \DTX{um-code-sym-commands.dtx}
18  \DTX{um-code-alphabets.dtx}
19  \DTX{um-code-primes.dtx}
20  \DTX{um-code-sscript.dtx}
21  \DTX{um-code-compat.dtx}
22  \DTX{um-code-amsmath.dtx}
23  \DTX{um-code-epilogue.dtx}
24 }
25 </dtx>
```

Now exit if we're using plain TeX when loading this file with unicode-math.ins.

```
26 <*dtx>
27 \ifx\plainoutput\undefined\else\expandafter\endinput\fi
28 </dtx>
```

Metadata for documentation; the title and authors of the package.

```
29 <*dtx>
30 \title{
31   Experimental Unicode mathematical typesetting:
32   The \pkg{unicode-math} package
33 }
34 \author{
35   \scshape Will Robertson\\
36   \itshape Philipp Stephani, Joseph Wright, Khaled Hosny, and others\\
37   \url{http://github.com/wspr/unicode-math}
```

```

38 }
39 </dtx>

```

Declare the package version and date.

```

40 <base>\RequirePackage{expl3}
41 <base>\ProvidesExplPackage{unicode-math}
42 <package&XE>\ProvidesExplPackage{unicode-math-xetex}
43 <package&LU>\ProvidesExplPackage{unicode-math-luatex}
44 <base|package> {2023/08/13} {0.8r} {Unicode maths in XeLaTeX and LuaLaTeX}

```

Here the version and date are setup for typesetting the documentation.

```

45 <*dtx>
46 \date{
47   \def\filedate{2023/08/13}
48   \def\fileversion{0.8r}
49   \filedate \quad \fileversion
50 }
51 </dtx>

```

2 *The unicode-math.sty loading file*

The `unicode-math.sty` file is a stub which loads necessary packages and then splits into a XeTeX- or LuaTeX-specific version of the package.

```

52 <base>\sys_if_engine_luatex:T
53 <base> {
54 <base>   \RequirePackageWithOptions{unicode-math-luatex}
55 <base>   \endinput
56 <base> }
57 <base>\sys_if_engine_xetex:T
58 <base> {
59 <base>   \RequirePackageWithOptions{unicode-math-xetex}
60 <base>   \endinput
61 <base> }
62 <base>\msg_new:nnn {unicode-math} {unsupported-engine}
63 <base> { Cannot~ be~ run~ with~ \c_sys_engine_str!\! Use~ XeLaTeX~ or~ Lu-
64   aLaTeX~ instead. }
64 <base>\msg_error:nn {unicode-math} {unsupported-engine}
65 <base>\endinput

```

File II

um-code-opening.dtx

3 *Start of the package code*

The prefix for unicode-math is um:

```
1 <@=um>
2 <*package>
```

Packages Assuming people are running up-to-date packages.

```
3 \RequirePackage{xparse,l3keys2e}
4 \RequirePackage{fontspec}
5 \RequirePackage{fix-cm}
6 \RequirePackage{amsmath}
7 <LU>\RequirePackage{lualatex-math}

8 \cs_set_protected:Npn \@@_after_package:nNn #1 #2 #3
9   {
10     \AtBeginDocument
11       {
12         \cs_new_protected:Npn #2 {#3}
13         \ifpackageloaded {#1} {#2} {}
14       }
15   }
```

3.1 *expl3 variants*

Variants needed from expl3:

```
16 \cs_set_protected_nopar:Npn \exp_last_unbraced:NNx { \::N \::x_unbraced \::: }
    For fontspec:
17 \cs_generate_variant:Nn \fontspec_set_family:Nnn {Nx,Nxx}
18 \cs_generate_variant:Nn \prop_get:NnNTF {cx}
19 \cs_generate_variant:Nn \tl_if_eq:nnF {o}
```

3.2 *Low level commands*

```
20 \cs_set_eq:NN \@@_group_begin: \group_begin:
21 \cs_set_protected:Npn \@@_group_end:n #1 { #1 \group_end: }
22 \cs_set_eq:NN \@@_group_begin_frozen: \@@_group_begin:
23 \cs_set_eq:NN \@@_group_end_frozen:n \@@_group_end:n
```

3.3 *Primitive font commands*

What might end up being provided by the kernel.

`\@@_glyph_if_exist:NnTF`

```
24 \prg_new_conditional:Nnn \@@_glyph_if_exist:Nn {p,TF,T,F}
25 {
26   \tex_iffontchar:D #1 #2 \scan_stop:
27   \prg_return_true:
28   \else:
29   \prg_return_false:
30   \fi:
31 }
```

`\@@_fontface_gset_eq:NN`

```
32 \cs_set_protected:Nn \@@_fontface_gset_eq:NN
33 {
34   \tex_global:D \tex_let:D #1 #2
35 }
36 \cs_generate_variant:Nn \@@_fontface_gset_eq:NN {cN}
```

3.3.1 *Mathcode and friends*

`\@@_set_mathcode:nnnn` These are all wrappers for the primitive commands that take numerical input only.

```
\@@_set_mathcode:nnn 37 \cs_set:Npn \@@_set_mathcode:nnnn #1#2#3#4
38 {
39   \Umathcode \int_eval:n {#1} =
40   \mathchar@type#2 \csname sym#3\endcsname \int_eval:n {#4} \scan_stop:
41 }
42 \cs_set:Npn \@@_set_mathcode:nnn #1#2#3
43 {
44   \Umathcode \int_eval:n {#1} =
45   \mathchar@type#2 \csname sym#3\endcsname \int_eval:n {#1} \scan_stop:
46 }
```

`\@@_set_mathchar:NNnn`

```
\@@_set_mathchar:cNnn 47 \cs_set:Npn \@@_set_mathchar:NNnn #1#2#3#4
48 {
49   \Umathchardef #1 =
50   \mathchar@type#2 \csname sym#3\endcsname \int_eval:n {#4} \scan_stop:
51 }
52 \cs_generate_variant:Nn \@@_set_mathchar:NNnn {c}
```

`\@@_set_delcode:nnn`

```
53 \cs_new:Nn \@@_set_delcode:nnn
54 {
55   \Udelcode#2 = \csname sym#1\endcsname #3 \scan_stop:
56 }
```

`\@@_radical:nn`

```
57 \cs_new:Nn \@@_radical:nn
58 {
```

```

59 \Uradical \csname sym#1\endcsname #2 \scan_stop:
60 }

```

`\@@_delimiter:Nnn`

```

61 \cs_new:Nn \@@_delimiter:Nnn
62 {
63   \Udelimiter \mathchar@type#1 \csname sym#2\endcsname #3 \scan_stop:
64 }

```

`\@@_accent:nnn`

```

65 \cs_new:Nn \@@_accent:nnn
66 {
67   \Umathaccent #1~ \mathchar@type\mathaccent \use:c { sym #2 } #3 \scan_stop:
68 }

```

`\@@_char_gmake_mathactive:N`

`\@@_char_gmake_mathactive:n`

```

69 \cs_new:Nn \@@_char_gmake_mathactive:N
70 {
71   \tex_global:D \tex_mathcode:D `#1 = "8000 \scan_stop:
72 }
73 \cs_new:Nn \@@_char_gmake_mathactive:n
74 {
75   \tex_global:D \tex_mathcode:D \int_eval:n {#1} = "8000 \scan_stop:
76 }

```

`\@@_mathactive_remap:nn` Makes #1 math-active and defines its meaning to be #2. This is a global operation.

```

77 \cs_new:Nn \@@_mathactive_remap:nn
78 {
79   \group_begin:
80     \cs_set_protected:Npn \@@_tmp: {#2}
81     \@@_char_gmake_mathactive:n {#1}
82     \char_gset_active_eq:nN {#1} \@@_tmp:
83   \group_end:
84 }

```

3.3.2 NFSS-related interfaces

`\@@_mathgroup_set:n` Remember that `\mathgroup` is just `\fam!`

```

85 \cs_new_protected:Nn \@@_mathgroup_set:n
86 {
87   \tex_fam:D #1 \scan_stop:
88 }

```

3.3.3 Font parameters

`\@@_copy_fontdimen:nnN`

```

89 \cs_new:Nn \@@_copy_fontdimen:nnN
90 {
91   \fontdimen #1 \font = \the \fontdimen #2 #3 \relax
92 }

```

`\@@_zero_fontdimen:n`

```
93 \cs_new:Nn \@@_zero_fontdimen:n
94 {
95   \fontdimen #1 \font = 0pt\relax
96 }
```

`\@@_int_if_zero_p:n`

`\@@_int_if_zero:nTF`

```
97 \prg_new_conditional:Nnn \@@_int_if_zero:n {p,TF,T,F}
98 {
99   \int_compare:nNnTF {#1} = 0 {\prg_return_true:} {\prg_return_false:}
100 }
```

3.4 *Alphabet Unicode positions (USVs)*

Before we begin, let's define the positions of the various Unicode alphabets so that our code is a little more readable.¹

`\usv_set:nnn` Rather than 'readable', in the end, this makes the code more extensible.

`\@@_to_usv:nn`

```
101 \cs_new:Nn \usv_set:nnn { \tl_const:cn { c_@@_#1_#2_usv } {#3} }
102 \cs_new:Nn \@@_to_usv:nn { \use:c { c_@@_#1_#2_usv } }
```

`\@@_usv_if_exist:nnTF`

```
103 \prg_new_conditional:Nnn \@@_usv_if_exist:nn {T,F,TF}
104 {
105   \cs_if_exist:cTF { c_@@_#1_#2_usv }
106     \prg_return_true: \prg_return_false:
107 }
```

3.5 *Overcoming \@onlypreamble*

The requirement of only setting up the maths fonts in the preamble is lifted. (Perhaps unwisely.)

`\tl_map_inline:nn`

```
108 {
109   \new@mathgroup\cdp@list\cdp@elt\DeclareMathSizes
110   \@DeclareMathSizes\newmathalphabet\newmathalphabet@@\newmathalphabet@@@
111   \DeclareMathVersion\define@mathalphabet\define@mathgroup\addtoversion
112   \version@list\version@elt\alpha@list\alpha@elt
113   \restore@mathversion\init@restore@version\dorestore@version\process@table
114   \new@mathversion\DeclareSymbolFont\group@list\group@elt
115   \new@symbolfont\SetSymbolFont\SetSymbolFont\get@cdp
116   \DeclareMathAlphabet\new@mathalphabet\SetMathAlphabet\SetMathAlphabet@
117   \DeclareMathAccent\set@mathaccent\DeclareMathSymbol\set@mathchar
118   \set@mathsymbol\DeclareMathDelimiter\@xxDeclareMathDelimiter
119   \@DeclareMathDelimiter\@xxDeclareMathDelimiter\set@mathdelimiter
120   \set@@mathdelimiter\DeclareMathRadical\mathchar@type
121   \DeclareSymbolFontAlphabet\DeclareSymbolFontAlphabet@
```

¹'u.s.v.' stands for 'Unicode scalar value'.

```

123 }
124 {
125   \tl_remove_once:Nn \@preamblecmds {\do#1}
126 }

```

3.6 Wrappers for kernel commands

Messages themselves are defined in section §8.

```

127 \cs_new:Npn \@_error:n { \msg_error:nn {unicode-math} }
128 \cs_new:Npn \@_error:nx { \msg_error:nxx {unicode-math} }
129 \cs_new:Npn \@_warning:n { \msg_warning:nn {unicode-math} }
130 \cs_new:Npn \@_warning:nnn { \msg_warning:nxxx {unicode-math} }
131 \cs_new:Npn \@_log:n { \msg_log:nn {unicode-math} }
132 \cs_new:Npn \@_log:nx { \msg_log:nxx {unicode-math} }

133 \cs_generate_variant:Nn \msg_new:nnn {nnx}
134 \cs_generate_variant:Nn \msg_new:nnnn {nxxx}
135 \cs_new:Nn \@_msg_new:nn { \msg_new:nxx {unicode-math} {#1} { \tl_trim_spaces:n {#2} } }

```

\@@_cs_new:Nn

```

136 (*debug)
137 \int_new:N \g_@@_debug_nest_int
138 \cs_new:Nn \@_debug:n
139 {
140   \typeout{ <UM~DEBUG>~\prg_replicate:nn \g_@@_debug_nest_int {::}~ #1}
141 }
142 \cs_new:Nn \@_debug_start:n
143 {
144   \int_gincr:N \g_@@_debug_nest_int
145   \@_debug:n {#1}
146 }
147 \cs_new:Nn \@_debug_end:n
148 {
149   \int_gdecr:N \g_@@_debug_nest_int
150 }
151 (/debug)

152 \cs_new:Npn \@_cs_set:Nn #1 #2
153 {
154   \cs_if_exist:NF #1 { \ERROR{CS~ DOES~ NOT~ EXIST,~ USE~ "NEW"} }
155   \cs_set_protected:Nn #1
156   {
157     (debug)\@@_debug_start:n { \cs_to_str:N #1 }
158     #2
159     (debug)\@@_debug_end:n { \cs_to_str:N #1 }
160   }
161 }
162 \cs_new:Npn \@_cs_new:Nn #1 #2
163 {
164   \cs_new_protected:Nn #1

```

```
165     {
166 <debug>\@@_debug_start:n { \cs_to_str:N #1 }
167     #2
168 <debug>\@@_debug_end:n   { \cs_to_str:N #1 }
169   }
170 }

171 </package>
```

File III

um-code-variables.dtx

4 Variable initialisation

1 *(*package)*

4.1 *bool*

True if using a proper OpenType font with unicode maths

2 `\bool_new:N \g_@@_ot_math_bool`

Set when `\setmathfont` is run to trap the problem of no main font defined.

3 `\bool_new:N \g_@@_main_font_defined_bool`

4 `\bool_new:N \g_@@_init_bool`

5 `\bool_new:N \l_@@_implicit_alph_bool`

For math-style:

6 `\bool_new:N \g_@@_literal_bool`

7 `\bool_new:N \g_@@_upLatin_bool`

8 `\bool_new:N \g_@@_uplatin_bool`

9 `\bool_new:N \g_@@_upGreek_bool`

10 `\bool_new:N \g_@@_upgreek_bool`

For bold-style:

11 `\bool_new:N \g_@@_bfliteral_bool`

12 `\bool_new:N \g_@@_bfupLatin_bool`

13 `\bool_new:N \g_@@_bfuplatin_bool`

14 `\bool_new:N \g_@@_bfupGreek_bool`

15 `\bool_new:N \g_@@_bfupgreek_bool`

For sans-style:

16 `\bool_new:N \g_@@_upsans_bool`

17 `\bool_new:N \g_@@_sfliteral_bool`

For assorted package options:

18 `\bool_new:N \g_@@_upNabla_bool`

19 `\bool_new:N \g_@@_uppartial_bool`

20 `\bool_new:N \g_@@_literal_Nabla_bool`

21 `\bool_new:N \g_@@_literal_partial_bool`

22 `\bool_new:N \l_@@_smallfrac_bool`

23 `\bool_new:N \g_@@_literal_colon_bool`

24 `\bool_new:N \g_@@_mathrm_text_bool`

25 `\bool_new:N \g_@@_mathit_text_bool`

26 `\bool_new:N \g_@@_mathbf_text_bool`

27 `\bool_new:N \g_@@_mathsf_text_bool`

28 `\bool_new:N \g_@@_mathtt_text_bool`

4.2 *int*

```
29 \int_new:N \g_@@_fam_int
30 \int_new:N \g_@@_fonts_used_int
31 \int_new:N \l_@@_primecount_int
```

4.3 *tl*

For displaying in warning messages, etc.:

```
32 \tl_const:Nn \c_@@_math_alphabet_name_latin_tl {Latin,~lowercase}
33 \tl_const:Nn \c_@@_math_alphabet_name_Latin_tl {Latin,~uppercase}
34 \tl_const:Nn \c_@@_math_alphabet_name_greek_tl {Greek,~lowercase}
35 \tl_const:Nn \c_@@_math_alphabet_name_Greek_tl {Greek,~uppercase}
36 \tl_const:Nn \c_@@_math_alphabet_name_num_tl {Numerals}
37 \tl_const:Nn \c_@@_math_alphabet_name_misc_tl {Misc.}

38 \tl_new:N \l_@@_style_tl
39 \tl_new:N \l_@@_family_tl
40 \tl_new:N \l_@@_alphabet_tl
41 \tl_new:N \l_@@_fontname_tl
42 \tl_new:N \l_@@_symfont_label_tl
43 \tl_new:N \l_@@_remap_style_tl
44 \tl_new:N \l_@@_fam_two_tl
45 \tl_new:N \l_@@_fam_three_tl
46 \tl_new:N \l_@@_curr_named_slot

47 \tl_new:N \l_@@_tmpa_tl
48 \tl_new:N \l_@@_tmpb_tl
49 \tl_new:N \l_@@_tmpc_tl
50 \tl_new:N \l_@@_mathstyle_tl
51 \tl_new:N \l_@@_radicals_tl
52 \tl_new:N \l_@@_nolimits_tl
53 \tl_new:N \l_@@_trial_family_tl
54 \tl_new:N \l_@@_ss_chain_tl
55 \tl_new:N \l_@@_tmpa_key_tl
```

Used to store the font switch for the `\operator@font`.

```
56 \tl_new:N \g_@@_operator_mathfont_tl

57 \tl_new:N \g_@@_slash_delimiter_usv
58 \tl_new:N \g_@@_mathparam_settings_tl
59 \tl_new:N \l_@@_mathtable_tl
60 \tl_new:N \g_@@_mathtable_tl
61 \tl_new:N \g_@@_fontname_tl
62 \tl_new:N \g_@@_mversion_tl
63 \tl_new:N \g_@@_symfont_tl
64 \tl_new:N \l_@@_font_keyval_tl
65 \tl_new:N \g_@@_family_tl
66 \tl_new:N \g_@@_style_tl
67 \tl_new:N \g_@@_remap_style_tl
68 \tl_new:N \l_@@_not_token_name_tl
69 \tl_new:N \g_@@_curr_font_cmd_tl
```

```

70 \tl_new:N \g_@@_sqrt_font_cmd_tl
71 \tl_new:N \g_@@_prime_font_cmd_tl

```

`\g_@@_mathparam_store_tl` Used to store and restore the math parameters used in LuaTeX. This is done to ‘save’ the values of the *first* (or main) maths font loaded, rather than (as per LuaTeX defaults) the last.

```

72 <*LU>
73 \tl_new:N \g_@@_mathparam_store_tl
74 </LU>

```

4.4 *clist*

```

75 \clist_new:N \g_@@_char_nrange_clist
76 \clist_new:N \g_@@_unknown_keys_clist
77 \clist_new:N \g_@@_alphabet_clist
78 \clist_new:N \l_@@_mathmap_charints_clist
79 \clist_new:N \l_@@_unknown_keys_clist
80 \clist_new:N \l_@@_keyval_clist
81 \clist_new:N \l_@@_alphabet_clist
82 \clist_new:N \g_@@_bad_alpha_clist
83 \clist_gput_right:Nx \g_@@_bad_alpha_clist { \tl_to_str:n {bf} }
84 \clist_gput_right:Nx \g_@@_bad_alpha_clist { \tl_to_str:n {sf} }
85 \clist_gput_right:Nx \g_@@_bad_alpha_clist { \tl_to_str:n {bfsf} }

```

4.5 *seq*

```

86 \seq_new:N \l_@@_missing_alph_seq
87 \seq_new:N \g_@@_mathalph_seq
88 \seq_new:N \g_@@_char_range_seq
89 \seq_new:N \g_@@_mclass_range_seq

```

`\g_@@_mathclasses_seq` Every math class.

```

90 \seq_new:N \g_@@_mathclasses_seq
91 \seq_gset_from_clist:Nn \g_@@_mathclasses_seq
92 {
93   \mathord, \mathalpha, \mathbin, \mathrel, \mathpunct,
94   \mathop,
95   \mathopen, \mathclose,
96   \mathfence, \mathover, \mathunder,
97   \mathaccent, \mathaccentoverlay, \mathbotaccent, \mathaccentwide, \mathbotaccentwide
98 }

```

`\g_@@_default_mathalph_seq` This sequence stores the alphabets in each math style.

```

99 \seq_new:N \g_@@_default_mathalph_seq

```

`\g_@@_mathstyles_seq` This is every ‘math style’ known to unicode-math. A named range is such as “bfit” and “sfit”, which are also math styles (with `\sybfit` and `\symsfit`). ‘Mathstyles’ are a superset of named ranges and also include commands such as `\symbf` and `\symsf`.

N.B. for parsing purposes ‘named ranges’ are defined as strings!

```

100 \seq_new:N \g_@@_mathstyles_seq

```


4.6 *prop*

101 \prop_new:N \g_@@_supers_prop

102 \prop_new:N \g_@@_subs_prop

4.7 *muskip*

103 \muskip_new:N \g_@@_primekern_muskip

104 \muskip_gset:Nn \g_@@_primekern_muskip { -\thinmuskip/2 }% arbitrary

4.8 *fp*

105 \fp_new:N \g_@@_size_tfsf_fp

106 \fp_new:N \g_@@_size_sfssf_fp

4.9 *quark*

`\q_unicode_math` Used as a flag within control sequences to check they're recognised by the package.

107 \quark_new:N \q_unicode_math

108 `</package>`

File IV

um-code-api.dtx

5 *Programmers' interface*

1 *<package>*

`\unimath_get_mathstyle:` This command expands to the currently math style.

2 `\cs_new:Nn \unimath_get_mathstyle:`

3 `{`

4 `\tl_use:N \l_@@_mathstyle_tl`

5 `}`

6 *</package>*

File V

um-code-ui.dtx

6 *The user interface commands*

1 *(*package)*

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

```
2 \NewDocumentCommand \unimathsetup {m} { \keys_set:nn {unicode-math} {#1} }
```

`\setmathfont[#1]`: font features (first optional argument retained for backwards compatibility)

#2 : font name

[#3]: font features

```
3 \NewDocumentCommand \setmathfont { O{} m O{} }
```

```
4 {
```

```
5   \@@_setmathfont:nn {#1,#3} {#2}
```

```
6 }
```

`\setmathfontface`

```
7 \NewDocumentCommand \setmathfontface { m O{} m O{} }
```

```
8 {
```

```
9   \@@_setmathfontface:Nnn #1 {#2,#4} {#3}
```

```
10 }
```

Note that L^AT_EX's `\SetMathAlphabet` simply doesn't work to "reset" a maths alphabet font after `\begin{document}`, so unlike most of the other maths commands around we still restrict this one to the preamble.

```
11 \onlypreamble \setmathfontface
```

`\setoperatorfont` TODO: add check?

```
12 \NewDocumentCommand \setoperatorfont {m}
```

```
13 {
```

```
14   \tl_gset:Nn \g_@@_operator_mathfont_tl {#1}
```

```
15 }
```

```
16 \setoperatorfont{\mathrm}
```

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

```
17 \NewDocumentCommand \addnolimits {m}
```

```
18 {
```

```
19   \tl_put_right:Nn \l_@@_nolimits_tl {#1}
```

```
20 }
```

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

```
21 \NewDocumentCommand \removenolimits {m}
```

```
22 {
```

```
23   \tl_remove_all:Nn \l_@@_nolimits_tl {#1}
```

```
24 }
```

25 `</package>`

File VI

um-code-pkgopt.dtx

7 *setup and package options*

1 *(*package)*

\@@_keys_choices:nn To simplify the creation of option keys, let's iterate in pairs rather than worry about equals signs and commas.

```
2 \cs_new:Nn \@@_keys_choices:nn
3 {
4   \cs_set:Npn \@@_keys_choices_fn:nn { \@@_keys_choices_aux:nnn {#1} }
5   \use:x
6   {
7     \exp_not:N \keys_define:nn {unicode-math}
8     {
9       #1 .choice: ,
10      \@@_tl_map_dbl:nN {#2} \@@_keys_choices_fn:nn
11    }
12  }
13 }
14 \cs_new:Nn \@@_keys_choices_aux:nnn { #1 / #2 .code:n = { \exp_not:n {#3} } , }
15 \cs_new:Nn \@@_tl_map_dbl:nN
16 {
17   \__@@_tl_map_dbl:Nnn #2 #1 \q_recursion_tail {}{} \q_recursion_stop
18 }
19 \cs_new:Nn \__@@_tl_map_dbl:Nnn
20 {
21   \quark_if_recursion_tail_stop:n {#2}
22   \quark_if_recursion_tail_stop:n {#3}
23   #1 {#2} {#3}
24   \__@@_tl_map_dbl:Nnn #1
25 }
```

Compatibility

```
26 \@@_keys_choices:nn {mathup}
27 {
28   {sym} { \bool_gset_false:N \g_@@_mathrm_text_bool }
29   {text} { \bool_gset_true:N \g_@@_mathrm_text_bool }
30 }
31 \@@_keys_choices:nn {mathrm}
32 {
33   {sym} { \bool_gset_false:N \g_@@_mathrm_text_bool }
34   {text} { \bool_gset_true:N \g_@@_mathrm_text_bool }
35 }
```

```

36 \@@_keys_choices:nn {mathit}
37 {
38   {sym} { \bool_gset_false:N \g_@@_mathit_text_bool }
39   {text} { \bool_gset_true:N \g_@@_mathit_text_bool }
40 }
41 \@@_keys_choices:nn {mathbf}
42 {
43   {sym} { \bool_gset_false:N \g_@@_mathbf_text_bool }
44   {text} { \bool_gset_true:N \g_@@_mathbf_text_bool }
45 }
46 \@@_keys_choices:nn {mathsf}
47 {
48   {sym} { \bool_gset_false:N \g_@@_mathsf_text_bool }
49   {text} { \bool_gset_true:N \g_@@_mathsf_text_bool }
50 }
51 \@@_keys_choices:nn {mathtt}
52 {
53   {sym} { \bool_gset_false:N \g_@@_mathtt_text_bool }
54   {text} { \bool_gset_true:N \g_@@_mathtt_text_bool }
55 }

```

math-style

```

56 \@@_keys_choices:nn {normal-style}
57 {
58   {ISO} {
59     \bool_gset_false:N \g_@@_literal_bool
60     \bool_gset_false:N \g_@@_upGreek_bool
61     \bool_gset_false:N \g_@@_upgreek_bool
62     \bool_gset_false:N \g_@@_upLatin_bool
63     \bool_gset_false:N \g_@@_uplatin_bool
64   }
65   {TeX} {
66     \bool_gset_false:N \g_@@_literal_bool
67     \bool_gset_true:N \g_@@_upGreek_bool
68     \bool_gset_false:N \g_@@_upgreek_bool
69     \bool_gset_false:N \g_@@_upLatin_bool
70     \bool_gset_false:N \g_@@_uplatin_bool
71   }
72   {french} {
73     \bool_gset_false:N \g_@@_literal_bool
74     \bool_gset_true:N \g_@@_upGreek_bool
75     \bool_gset_true:N \g_@@_upgreek_bool
76     \bool_gset_true:N \g_@@_upLatin_bool
77     \bool_gset_false:N \g_@@_uplatin_bool
78   }
79   {upright} {
80     \bool_gset_false:N \g_@@_literal_bool
81     \bool_gset_true:N \g_@@_upGreek_bool

```

```

82         \bool_gset_true:N \g_@@_upgreek_bool
83         \bool_gset_true:N \g_@@_upLatin_bool
84         \bool_gset_true:N \g_@@_uplatin_bool
85     }
86     {literal} {
87         \bool_gset_true:N \g_@@_literal_bool
88     }
89 }
90 \@@_keys_choices:nn {math-style}
91 {
92     {ISO} {
93         \unimathsetup { nabla=upright, partial=italic,
94             normal-style=ISO, bold-style=ISO, sans-style=italic }
95     }
96     {TeX} {
97         \unimathsetup { nabla=upright, partial=italic,
98             normal-style=TeX, bold-style=TeX, sans-style=upright }
99     }
100    {french} {
101        \unimathsetup { nabla=upright, partial=upright,
102            normal-style=french, bold-style=upright, sans-style=upright }
103    }
104    {upright} {
105        \unimathsetup { nabla=upright, partial=upright,
106            normal-style=upright, bold-style=upright, sans-style=upright }
107    }
108    {literal} {
109        \unimathsetup { colon=literal, nabla=literal, partial=literal,
110            normal-style=literal, bold-style=literal, sans-style=literal }
111    }
112 }

```

bold-style

```

113 \@@_keys_choices:nn {bold-style}
114 {
115     {ISO} {
116         \bool_gset_false:N \g_@@_bfliteral_bool
117         \bool_gset_false:N \g_@@_bfupGreek_bool
118         \bool_gset_false:N \g_@@_bfupgreek_bool
119         \bool_gset_false:N \g_@@_bfupLatin_bool
120         \bool_gset_false:N \g_@@_bfuplatin_bool
121     }
122     {TeX} {
123         \bool_gset_false:N \g_@@_bfliteral_bool
124         \bool_gset_true:N \g_@@_bfupGreek_bool
125         \bool_gset_false:N \g_@@_bfupgreek_bool
126         \bool_gset_true:N \g_@@_bfupLatin_bool
127         \bool_gset_true:N \g_@@_bfuplatin_bool

```

```

128     }
129   {upright} {
130     \bool_gset_false:N \g_@@_bfliteral_bool
131     \bool_gset_true:N \g_@@_bfupGreek_bool
132     \bool_gset_true:N \g_@@_bfupgreek_bool
133     \bool_gset_true:N \g_@@_bfupLatin_bool
134     \bool_gset_true:N \g_@@_bfuplatin_bool
135   }
136   {literal} {
137     \bool_gset_true:N \g_@@_bfliteral_bool
138   }
139 }

```

sans-style

```

140 \@@_keys_choices:nn {sans-style}
141 {
142   {italic} { \bool_gset_false:N \g_@@_upsans_bool }
143   {upright} { \bool_gset_true:N \g_@@_upsans_bool }
144   {literal} { \bool_gset_true:N \g_@@_sfliteral_bool }
145 }

```

Nabla and partial

```

146 \@@_keys_choices:nn {nabla}
147 {
148   {upright} {
149     \bool_gset_false:N \g_@@_literal_Nabla_bool
150     \bool_gset_true:N \g_@@_upNabla_bool
151   }
152   {italic} {
153     \bool_gset_false:N \g_@@_literal_Nabla_bool
154     \bool_gset_false:N \g_@@_upNabla_bool
155   }
156   {literal} {
157     \bool_gset_true:N \g_@@_literal_Nabla_bool
158   }
159 }
160 \@@_keys_choices:nn {partial}
161 {
162   {upright} {
163     \bool_gset_false:N \g_@@_literal_partial_bool
164     \bool_gset_true:N \g_@@_uppartial_bool
165   }
166   {italic} {
167     \bool_gset_false:N \g_@@_literal_partial_bool
168     \bool_gset_false:N \g_@@_uppartial_bool
169   }
170   {literal} {
171     \bool_gset_true:N \g_@@_literal_partial_bool

```



```

172     }
173 }

```

Colon style

```

174 \@@_keys_choices:nn {colon}
175 {
176   {literal} { \bool_gset_true:N \g_@@_literal_colon_bool }
177   {TeX}     { \bool_gset_false:N \g_@@_literal_colon_bool }
178 }

```

Slash delimiter style

```

179 \@@_keys_choices:nn {slash-delimiter}
180 {
181   {ascii} { \tl_gset:Nn \g_@@_slash_delimiter_usv {"002F} }
182   {frac}  { \tl_gset:Nn \g_@@_slash_delimiter_usv {"2044} }
183   {div}   { \tl_gset:Nn \g_@@_slash_delimiter_usv {"2215} }
184 }

```

Active fraction style

```

185 \@@_keys_choices:nn {active-frac}
186 {
187   {small}
188   {
189     \bool_set_true:N \l_@@_smallfrac_bool
190     \use:c {@@_setup_active_frac:}
191   }
192
193   {normalsize}
194   {
195     \bool_set_false:N \l_@@_smallfrac_bool
196     \use:c {@@_setup_active_frac:}
197   }
198 }

```

Debug/tracing

```

199 \keys_define:nn {unicode-math}
200 {
201   warnings-off .code:n =
202   {
203     \clist_map_inline:nn {#1}
204     { \msg_redirect_name:nnn { unicode-math } { ##1 } { none } }
205   }
206 }
207 \@@_keys_choices:nn {trace}
208 {
209   {on}   {} % default
210   {debug} { \msg_redirect_module:nnn { unicode-math } { log } { warning } }

```

```
211 {off} { \msg_redirect_module:nnn { unicode-math } { log } { none } }  
212 }
```

7.1 Defaults

```
213 \unimathsetup {math-style=TeX}  
214 \unimathsetup {slash-delimiter=ascii}  
215 \unimathsetup {trace=off}  
216 \unimathsetup {mathrm=text,mathit=text,mathbf=text,mathsf=text,mathtt=text}  
217 \cs_if_exist:NT \tfrac { \unimathsetup {active-frac=small} }  
218 \ProcessKeysOptions {unicode-math}  
219 </package>
```

File VII

um-code-msg.dtx

8 *Error messages*

```
1 (*package)
2 \char_set_catcode_space:n {32}
3 \@@_msg_new:nn {default-math-font}
4 {
5   Defining the default maths font as '\l_@@_fontname_tl'.
6 }
7 \@@_msg_new:nn {setup-implicit}
8 {
9   Setup alphabets: implicit mode.
10 }
11 \@@_msg_new:nn {setup-explicit}
12 {
13   Setup alphabets: explicit mode.
14 }
15 \@@_msg_new:nn {alph-initialise}
16 {
17   Initialising \@backslashchar math#1.
18 }
19 \@@_msg_new:nn {setup-alph}
20 {
21   Setup alphabet: #1.
22 }
23 \@@_msg_new:nn {no-alphabet}
24 {
25   I am trying to set up alphabet"#1" but there are no configuration set-
26   tings for it.
27   (See source file "unicode-math-alphabets.dtx" to debug.)
28 }
29 \@@_msg_new:nn {no-named-range}
30 {
31   I am trying to define new alphabet "#2" in range "#1", but range "#1" hasn't been de-
32   fined yet.
33 }
34 \@@_msg_new:nn {missing-alphabets}
35 {
36   Missing math alphabets in font "\fontname\g_@@_curr_font_cmd_tl" \ \ \
37   \seq_map_function:NN \l_@@_missing_alph_seq \@@_print_indent:n
38 }
39 \cs_new:Nn \@@_print_indent:n { \space\space\space\space #1 \ \ }
40 \@@_msg_new:nn {macro-expected}
41 {
42   I've expected that #1 is a macro, but it isn't.
```

```

41 }
42 \@@_msg_new:nn {wrong-meaning}
43 {
44   I've expected #1 to have the meaning #3, but it has the meaning #2.
45 }
46 \@@_msg_new:nn {patch-macro}
47 {
48   I'm going to patch macro #1.
49 }
50 \@@_msg_new:nn {mathtools-overbracket} {
51   Using \token_to_str:N \overbracket\ and
52     \token_to_str:N \underbracket\ from
53   'mathtools' package.\
54   \
55   Use \token_to_str:N \Uoverbracket\ and
56     \token_to_str:N \Underbracket\ for
57   original 'unicode-math' definition.
58 }
59 \@@_msg_new:nn {mathtools-colon} {
60   I'm going to overwrite the following commands from
61   the 'mathtools' package: \
62   \ \ \ \ \token_to_str:N \dblcolon,
63   \token_to_str:N \coloneqq,
64   \token_to_str:N \Coloneqq,
65   \token_to_str:N \eqqcolon. \
66   Note that since I won't overwrite the other colon-like
67   commands, using them will lead to inconsistencies.
68 }
69 \@@_msg_new:nn {colonequals} {
70   I'm going to overwrite the following commands from
71   the 'colonequals' package: \
72   \ \ \ \ \token_to_str:N \ratio,
73     \token_to_str:N \coloncolon,
74     \token_to_str:N \minuscolon, \
75   \ \ \ \ \token_to_str:N \colonequals,
76     \token_to_str:N \equalscolon,
77     \token_to_str:N \coloncolonequals. \
78   Note that since I won't overwrite the other colon-like
79   commands, using them will lead to inconsistencies.
80   Furthermore, changing \token_to_str:N \colonsep \c_space_tl
81   or \token_to_str:N \doublecolonsep \c_space_tl won't have
82   any effect on the re-defined commands.
83 }
84 \@@_msg_new:nn {bad-cs-in-range}
85 {
86   Command '#1' in math range is not recognised as a maths symbol.
87   Check file "unicode-math-table.tex" for allowable commands.
88 }
89 \@@_msg_new:nn {legacy-char-not-supported}

```

```

90 {
91   Command `#1` is a legacy maths symbol that is not supported by unicode-
math.
92 }
93 \@_msg_new:nn {range-not-bf-sf}
94 {
95   Range alphabets cannot include alphabets referring to `bf`, `sf`, or `bfsf`
96   since they relate to input commands not output glyphs.
97   Use `bfit` or `bfup` (etc.) to specify which.
98 }
99 \@_msg_new:nn {no-main-font}
100 {
101   No main maths font has been set up yet.\\If you simply want ‘the de-
fault’, use: \\
102   \iow_indent:n {\token_to_str:N\setmathfont{latinmodern-math.otf}}
103 }
104 \@_msg_new:nn {not-ot-math}
105 {
106   The first font loaded by unicode-math must be an OpenType Math font (with script=math).
107   If you simply want ‘the default’ before loading supplemen-
tary fonts over the top for certain
108   ranges, use: \\
109   \iow_indent:n {\token_to_str:N\setmathfont{latinmodern-math.otf}}
110 }
111 \char_set_catcode_ignore:n {32}
112 </package>

```

File VIII

um-code-usv.dtx

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.²

```
1 (*package)
```

Alphabets 'Normal':

```
2 \usv_set:nnn {normal} {num}      {48}
3 \usv_set:nnn {normal} {Latin}    {"1D434}
4 \usv_set:nnn {normal} {latin}    {"1D44E}
5 \usv_set:nnn {normal} {Greek}    {"1D6E2}
6 \usv_set:nnn {normal} {greek}    {"1D6FC}
7 \usv_set:nnn {normal} {varTheta} {"1D6F3}
8 \usv_set:nnn {normal} {epsilon}  {"1D716}
9 \usv_set:nnn {normal} {vartheta} {"1D717}
10 \usv_set:nnn {normal} {varkappa} {"1D718}
11 \usv_set:nnn {normal} {phi}      {"1D719}
12 \usv_set:nnn {normal} {varrho}   {"1D71A}
13 \usv_set:nnn {normal} {varpi}    {"1D71B}
14 \usv_set:nnn {normal} {Nabla}    {"1D6FB}
15 \usv_set:nnn {normal} {partial}  {"1D715}
```

Regular weights:

```
16 \usv_set:nnn {up}   {num}   {48}
17 \usv_set:nnn {up}   {Latin}  {65}
18 \usv_set:nnn {up}   {latin}  {97}
19 \usv_set:nnn {up}   {Greek}  {"391}
20 \usv_set:nnn {up}   {greek}  {"3B1}
21 \usv_set:nnn {it}   {Latin}  {"1D434}
22 \usv_set:nnn {it}   {latin}  {"1D44E}
23 \usv_set:nnn {it}   {Greek}  {"1D6E2}
24 \usv_set:nnn {it}   {greek}  {"1D6FC}
25 \usv_set:nnn {bb}   {num}    {"1D7D8}
26 \usv_set:nnn {bb}   {Latin}  {"1D538}
27 \usv_set:nnn {bb}   {latin}  {"1D552}
28 \usv_set:nnn {scr}  {Latin}  {"1D49C}
29 \usv_set:nnn {cal}  {Latin}  {"1D49C}
30 \usv_set:nnn {scr}  {latin}  {"1D4B6}
31 \usv_set:nnn {cal}  {latin}  {"1D4B6}
32 \usv_set:nnn {frak} {Latin}  {"1D504}
33 \usv_set:nnn {frak} {latin}  {"1D51E}
34 \usv_set:nnn {sf}   {num}    {"1D7E2}
```

²'u.s.v.' stands for 'Unicode scalar value'.

35 \usv_set:nnn {sfup} {num} {"1D7E2}
 36 \usv_set:nnn {sfit} {num} {"1D7E2}
 37 \usv_set:nnn {sfup} {Latin} {"1D5A0}
 38 \usv_set:nnn {sf} {Latin} {"1D5A0}
 39 \usv_set:nnn {sfup} {latin} {"1D5BA}
 40 \usv_set:nnn {sf} {latin} {"1D5BA}
 41 \usv_set:nnn {sfit} {Latin} {"1D608}
 42 \usv_set:nnn {sfit} {latin} {"1D622}
 43 \usv_set:nnn {tt} {num} {"1D7F6}
 44 \usv_set:nnn {tt} {Latin} {"1D670}
 45 \usv_set:nnn {tt} {latin} {"1D68A}

Bold weights:

46 \usv_set:nnn {bf} {num} {"1D7CE}
 47 \usv_set:nnn {bfup} {num} {"1D7CE}
 48 \usv_set:nnn {bfit} {num} {"1D7CE}
 49 \usv_set:nnn {bfup} {Latin} {"1D400}
 50 \usv_set:nnn {bfup} {latin} {"1D41A}
 51 \usv_set:nnn {bfup} {Greek} {"1D6A8}
 52 \usv_set:nnn {bfup} {greek} {"1D6C2}
 53 \usv_set:nnn {bfit} {Latin} {"1D468}
 54 \usv_set:nnn {bfit} {latin} {"1D482}
 55 \usv_set:nnn {bfit} {Greek} {"1D71C}
 56 \usv_set:nnn {bfit} {greek} {"1D736}
 57 \usv_set:nnn {bffrak} {Latin} {"1D56C}
 58 \usv_set:nnn {bffrak} {latin} {"1D586}
 59 \usv_set:nnn {bfscr} {Latin} {"1D4D0}
 60 \usv_set:nnn {bfcal} {Latin} {"1D4D0}
 61 \usv_set:nnn {bfscr} {latin} {"1D4EA}
 62 \usv_set:nnn {bfcal} {latin} {"1D4EA}
 63 \usv_set:nnn {bfsf} {num} {"1D7EC}
 64 \usv_set:nnn {bfsfup} {num} {"1D7EC}
 65 \usv_set:nnn {bfsfit} {num} {"1D7EC}
 66 \usv_set:nnn {bfsfup} {Latin} {"1D5D4}
 67 \usv_set:nnn {bfsfup} {latin} {"1D5EE}
 68 \usv_set:nnn {bfsfup} {Greek} {"1D756}
 69 \usv_set:nnn {bfsfup} {greek} {"1D770}
 70 \usv_set:nnn {bfsfit} {Latin} {"1D63C}
 71 \usv_set:nnn {bfsfit} {latin} {"1D656}
 72 \usv_set:nnn {bfsfit} {Greek} {"1D790}
 73 \usv_set:nnn {bfsfit} {greek} {"1D7AA}

The 'auto' bolds:

74 \usv_set:nnn {bfsf} {Latin} { \bool_if:NTF \g_@@_upLatin_bool \g_@@_bfsfup_Latin_usv \g_@@_bfsfi
 75 \usv_set:nnn {bfsf} {latin} { \bool_if:NTF \g_@@_uplatin_bool \g_@@_bfsfup_latin_usv \g_@@_bfsfi
 76 \usv_set:nnn {bfsf} {Greek} { \bool_if:NTF \g_@@_upGreek_bool \g_@@_bfsfup_Greek_usv \g_@@_bfsfi
 77 \usv_set:nnn {bfsf} {greek} { \bool_if:NTF \g_@@_upgreek_bool \g_@@_bfsfup_greek_usv \g_@@_bfsfi
 78 \usv_set:nnn {bf} {Latin} { \bool_if:NTF \g_@@_bfupLatin_bool \g_@@_bfup_Latin_usv \g_@@_bfit_L
 79 \usv_set:nnn {bf} {latin} { \bool_if:NTF \g_@@_bfuplatin_bool \g_@@_bfup_latin_usv \g_@@_bfit_L
 80 \usv_set:nnn {bf} {Greek} { \bool_if:NTF \g_@@_bfupGreek_bool \g_@@_bfup_Greek_usv \g_@@_bfit_L

81 \usv_set:nnn {bf} {greek} { \bool_if:NTF \g_@@_bfupgreek_bool \g_@@_bfup_greek_usv \g_@@_bfit_g

Greek variants Upright:

82 \usv_set:nnn {up} {varTheta} {"3F4}
83 \usv_set:nnn {up} {Digamma} {"3DC}
84 \usv_set:nnn {up} {epsilon} {"3F5}
85 \usv_set:nnn {up} {vartheta} {"3D1}
86 \usv_set:nnn {up} {varkappa} {"3F0}
87 \usv_set:nnn {up} {phi} {"3D5}
88 \usv_set:nnn {up} {varrho} {"3F1}
89 \usv_set:nnn {up} {varpi} {"3D6}
90 \usv_set:nnn {up} {digamma} {"3DD}

Bold:

91 \usv_set:nnn {bfup} {varTheta} {"1D6B9}
92 \usv_set:nnn {bfup} {Digamma} {"1D7CA}
93 \usv_set:nnn {bfup} {epsilon} {"1D6DC}
94 \usv_set:nnn {bfup} {vartheta} {"1D6DD}
95 \usv_set:nnn {bfup} {varkappa} {"1D6DE}
96 \usv_set:nnn {bfup} {phi} {"1D6DF}
97 \usv_set:nnn {bfup} {varrho} {"1D6E0}
98 \usv_set:nnn {bfup} {varpi} {"1D6E1}
99 \usv_set:nnn {bfup} {digamma} {"1D7CB}

Italic:

100 \usv_set:nnn {it} {varTheta} {"1D6F3}
101 \usv_set:nnn {it} {epsilon} {"1D716}
102 \usv_set:nnn {it} {vartheta} {"1D717}
103 \usv_set:nnn {it} {varkappa} {"1D718}
104 \usv_set:nnn {it} {phi} {"1D719}
105 \usv_set:nnn {it} {varrho} {"1D71A}
106 \usv_set:nnn {it} {varpi} {"1D71B}

Bold italic:

107 \usv_set:nnn {bfit} {varTheta} {"1D72D}
108 \usv_set:nnn {bfit} {epsilon} {"1D750}
109 \usv_set:nnn {bfit} {vartheta} {"1D751}
110 \usv_set:nnn {bfit} {varkappa} {"1D752}
111 \usv_set:nnn {bfit} {phi} {"1D753}
112 \usv_set:nnn {bfit} {varrho} {"1D754}
113 \usv_set:nnn {bfit} {varpi} {"1D755}

Bold sans:

114 \usv_set:nnn {bfsfup} {varTheta} {"1D767}
115 \usv_set:nnn {bfsfup} {epsilon} {"1D78A}
116 \usv_set:nnn {bfsfup} {vartheta} {"1D78B}
117 \usv_set:nnn {bfsfup} {varkappa} {"1D78C}
118 \usv_set:nnn {bfsfup} {phi} {"1D78D}
119 \usv_set:nnn {bfsfup} {varrho} {"1D78E}
120 \usv_set:nnn {bfsfup} {varpi} {"1D78F}

Bold sans italic:

121 \usv_set:nnn {bfsfit} {varTheta} {"1D7A1}
122 \usv_set:nnn {bfsfit} {epsilon} {"1D7C4}
123 \usv_set:nnn {bfsfit} {vartheta} {"1D7C5}
124 \usv_set:nnn {bfsfit} {varkappa} {"1D7C6}
125 \usv_set:nnn {bfsfit} {phi} {"1D7C7}
126 \usv_set:nnn {bfsfit} {varrho} {"1D7C8}
127 \usv_set:nnn {bfsfit} {varpi} {"1D7C9}

Nabla:

128 \usv_set:nnn {up} {Nabla} {"02207}
129 \usv_set:nnn {it} {Nabla} {"1D6FB}
130 \usv_set:nnn {bfup} {Nabla} {"1D6C1}
131 \usv_set:nnn {bfit} {Nabla} {"1D735}
132 \usv_set:nnn {bfsfup} {Nabla} {"1D76F}
133 \usv_set:nnn {bfsfit} {Nabla} {"1D7A9}

Partial:

134 \usv_set:nnn {up} {partial} {"02202}
135 \usv_set:nnn {it} {partial} {"1D715}
136 \usv_set:nnn {bfup} {partial} {"1D6DB}
137 \usv_set:nnn {bfit} {partial} {"1D74F}
138 \usv_set:nnn {bfsfup} {partial} {"1D789}
139 \usv_set:nnn {bfsfit} {partial} {"1D7C3}

Exceptions Upright uppercase:

140 \usv_set:nnn {up} {B} {\`B}
141 \usv_set:nnn {up} {C} {\`C}
142 \usv_set:nnn {up} {D} {\`D}
143 \usv_set:nnn {up} {E} {\`E}
144 \usv_set:nnn {up} {F} {\`F}
145 \usv_set:nnn {up} {H} {\`H}
146 \usv_set:nnn {up} {I} {\`I}
147 \usv_set:nnn {up} {L} {\`L}
148 \usv_set:nnn {up} {M} {\`M}
149 \usv_set:nnn {up} {N} {\`N}
150 \usv_set:nnn {up} {P} {\`P}
151 \usv_set:nnn {up} {Q} {\`Q}
152 \usv_set:nnn {up} {R} {\`R}
153 \usv_set:nnn {up} {Z} {\`Z}

Italic uppercase:

154 \usv_set:nnn {it} {B} {"1D435}
155 \usv_set:nnn {it} {C} {"1D436}
156 \usv_set:nnn {it} {D} {"1D437}
157 \usv_set:nnn {it} {E} {"1D438}
158 \usv_set:nnn {it} {F} {"1D439}
159 \usv_set:nnn {it} {H} {"1D43B}
160 \usv_set:nnn {it} {I} {"1D43C}
161 \usv_set:nnn {it} {L} {"1D43F}

162 \usv_set:nnn {it} {M} {"1D440}
 163 \usv_set:nnn {it} {N} {"1D441}
 164 \usv_set:nnn {it} {P} {"1D443}
 165 \usv_set:nnn {it} {Q} {"1D444}
 166 \usv_set:nnn {it} {R} {"1D445}
 167 \usv_set:nnn {it} {Z} {"1D44D}

Upright lowercase (needed for later mappings):

168 \usv_set:nnn {up} {d} {\`d}
 169 \usv_set:nnn {up} {e} {\`e}
 170 \usv_set:nnn {up} {g} {\`g}
 171 \usv_set:nnn {up} {h} {\`h}
 172 \usv_set:nnn {up} {i} {\`i}
 173 \usv_set:nnn {up} {j} {\`j}
 174 \usv_set:nnn {up} {o} {\`o}

Italic lowercase:

175 \usv_set:nnn {it} {d} {"1D451}
 176 \usv_set:nnn {it} {e} {"1D452}
 177 \usv_set:nnn {it} {g} {"1D454}
 178 \usv_set:nnn {it} {h} {"0210E}
 179 \usv_set:nnn {it} {i} {"1D456}
 180 \usv_set:nnn {it} {j} {"1D457}
 181 \usv_set:nnn {it} {o} {"1D45C}

Latin 'h':

182 \usv_set:nnn {bb} {h} {"1D559}
 183 \usv_set:nnn {tt} {h} {"1D691}
 184 \usv_set:nnn {scr} {h} {"1D4BD}
 185 \usv_set:nnn {cal} {h} {"1D4BD}
 186 \usv_set:nnn {frak} {h} {"1D525}
 187 \usv_set:nnn {bfup} {h} {"1D421}
 188 \usv_set:nnn {bfit} {h} {"1D489}
 189 \usv_set:nnn {sfup} {h} {"1D5C1}
 190 \usv_set:nnn {sfit} {h} {"1D629}
 191 \usv_set:nnn {bffrak} {h} {"1D58D}
 192 \usv_set:nnn {bfscr} {h} {"1D4F1}
 193 \usv_set:nnn {bfcal} {h} {"1D4F1}
 194 \usv_set:nnn {bfsfup} {h} {"1D5F5}
 195 \usv_set:nnn {bfsfit} {h} {"1D65D}

Dotless 'i' and 'j':

196 \usv_set:nnn {up} {dotlessi} {"00131}
 197 \usv_set:nnn {up} {dotlessj} {"00237}
 198 \usv_set:nnn {it} {dotlessi} {"1D6A4}
 199 \usv_set:nnn {it} {dotlessj} {"1D6A5}

Blackboard:

200 \usv_set:nnn {bb} {C} {"2102}
 201 \usv_set:nnn {bb} {H} {"210D}
 202 \usv_set:nnn {bb} {N} {"2115}

203 \usv_set:nnn {bb} {P} {"2119}
 204 \usv_set:nnn {bb} {Q} {"211A}
 205 \usv_set:nnn {bb} {R} {"211D}
 206 \usv_set:nnn {bb} {Z} {"2124}
 207 \usv_set:nnn {up} {Pi} {"003A0}
 208 \usv_set:nnn {up} {pi} {"003C0}
 209 \usv_set:nnn {up} {Gamma} {"00393}
 210 \usv_set:nnn {up} {gamma} {"003B3}
 211 \usv_set:nnn {up} {summation} {"02211}
 212 \usv_set:nnn {it} {Pi} {"1D6F1}
 213 \usv_set:nnn {it} {pi} {"1D70B}
 214 \usv_set:nnn {it} {Gamma} {"1D6E4}
 215 \usv_set:nnn {it} {gamma} {"1D6FE}
 216 \usv_set:nnn {bb} {Pi} {"0213F}
 217 \usv_set:nnn {bb} {pi} {"0213C}
 218 \usv_set:nnn {bb} {Gamma} {"0213E}
 219 \usv_set:nnn {bb} {gamma} {"0213D}
 220 \usv_set:nnn {bb} {summation} {"02140}

Italic blackboard:

221 \usv_set:nnn {bbit} {D} {"2145}
 222 \usv_set:nnn {bbit} {d} {"2146}
 223 \usv_set:nnn {bbit} {e} {"2147}
 224 \usv_set:nnn {bbit} {i} {"2148}
 225 \usv_set:nnn {bbit} {j} {"2149}

Script:

226 \usv_set:nnn {scr} {B} {"212C}
 227 \usv_set:nnn {scr} {E} {"2130}
 228 \usv_set:nnn {scr} {F} {"2131}
 229 \usv_set:nnn {scr} {H} {"210B}
 230 \usv_set:nnn {scr} {I} {"2110}
 231 \usv_set:nnn {scr} {L} {"2112}
 232 \usv_set:nnn {scr} {M} {"2133}
 233 \usv_set:nnn {scr} {R} {"211B}
 234 \usv_set:nnn {scr} {e} {"212F}
 235 \usv_set:nnn {scr} {g} {"210A}
 236 \usv_set:nnn {scr} {o} {"2134}

Calligraphic:

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

Fraktur:

```
248 \usv_set:nnn {frac} {C} {"212D}
249 \usv_set:nnn {frac} {H} {"210C}
250 \usv_set:nnn {frac} {I} {"2111}
251 \usv_set:nnn {frac} {R} {"211C}
252 \usv_set:nnn {frac} {Z} {"2128}
253 </package>
```

9.1 STIX fonts

Version 1.0.0 of the STIX fonts contains a number of alphabets in the private use area of Unicode; i.e., it contains many math glyphs that have not (yet or if ever) been accepted into the Unicode standard.

But we still want to be able to use them if possible.

```
254 <*stix>
```

Upright

```
255 \usv_set:nnn {stixsfup}{partial}{"E17C}
256 \usv_set:nnn {stixsfup}{Greek}{"E17D}
257 \usv_set:nnn {stixsfup}{greek}{"E196}
258 \usv_set:nnn {stixsfup}{varTheta}{"E18E}
259 \usv_set:nnn {stixsfup}{epsilon}{"E1AF}
260 \usv_set:nnn {stixsfup}{vartheta}{"E1B0}
261 \usv_set:nnn {stixsfup}{varkappa}{0000} % ???
262 \usv_set:nnn {stixsfup}{phi}{"E1B1}
263 \usv_set:nnn {stixsfup}{varrho}{"E1B2}
264 \usv_set:nnn {stixsfup}{varpi}{"E1B3}
265 \usv_set:nnn {stixupslash}{Greek}{"E2FC}
```

Italic

```
266 \usv_set:nnn {stixbbit}{A}{"E154}
267 \usv_set:nnn {stixbbit}{B}{"E155}
268 \usv_set:nnn {stixbbit}{E}{"E156}
269 \usv_set:nnn {stixbbit}{F}{"E157}
270 \usv_set:nnn {stixbbit}{G}{"E158}
271 \usv_set:nnn {stixbbit}{I}{"E159}
272 \usv_set:nnn {stixbbit}{J}{"E15A}
273 \usv_set:nnn {stixbbit}{K}{"E15B}
274 \usv_set:nnn {stixbbit}{L}{"E15C}
275 \usv_set:nnn {stixbbit}{M}{"E15D}
276 \usv_set:nnn {stixbbit}{O}{"E15E}
277 \usv_set:nnn {stixbbit}{S}{"E15F}
278 \usv_set:nnn {stixbbit}{T}{"E160}
279 \usv_set:nnn {stixbbit}{U}{"E161}
280 \usv_set:nnn {stixbbit}{V}{"E162}
281 \usv_set:nnn {stixbbit}{W}{"E163}
282 \usv_set:nnn {stixbbit}{X}{"E164}
```

283 \usv_set:nnn {stixbbit}{Y}{E165}
284 \usv_set:nnn {stixbbit}{a}{E166}
285 \usv_set:nnn {stixbbit}{b}{E167}
286 \usv_set:nnn {stixbbit}{c}{E168}
287 \usv_set:nnn {stixbbit}{f}{E169}
288 \usv_set:nnn {stixbbit}{g}{E16A}
289 \usv_set:nnn {stixbbit}{h}{E16B}
290 \usv_set:nnn {stixbbit}{k}{E16C}
291 \usv_set:nnn {stixbbit}{l}{E16D}
292 \usv_set:nnn {stixbbit}{m}{E16E}
293 \usv_set:nnn {stixbbit}{n}{E16F}
294 \usv_set:nnn {stixbbit}{o}{E170}
295 \usv_set:nnn {stixbbit}{p}{E171}
296 \usv_set:nnn {stixbbit}{q}{E172}
297 \usv_set:nnn {stixbbit}{r}{E173}
298 \usv_set:nnn {stixbbit}{s}{E174}
299 \usv_set:nnn {stixbbit}{t}{E175}
300 \usv_set:nnn {stixbbit}{u}{E176}
301 \usv_set:nnn {stixbbit}{v}{E177}
302 \usv_set:nnn {stixbbit}{w}{E178}
303 \usv_set:nnn {stixbbit}{x}{E179}
304 \usv_set:nnn {stixbbit}{y}{E17A}
305 \usv_set:nnn {stixbbit}{z}{E17B}

306 \usv_set:nnn {stixsfit}{Numerals}{E1B4}
307 \usv_set:nnn {stixsfit}{partial}{E1BE}
308 \usv_set:nnn {stixsfit}{Greek}{E1BF}
309 \usv_set:nnn {stixsfit}{greek}{E1D8}
310 \usv_set:nnn {stixsfit}{varTheta}{E1D0}
311 \usv_set:nnn {stixsfit}{epsilon}{E1F1}
312 \usv_set:nnn {stixsfit}{vartheta}{E1F2}
313 \usv_set:nnn {stixsfit}{varkappa}{0000} % ???
314 \usv_set:nnn {stixsfit}{phi}{E1F3}
315 \usv_set:nnn {stixsfit}{varrho}{E1F4}
316 \usv_set:nnn {stixsfit}{varpi}{E1F5}

317 \usv_set:nnn {stixcal}{Latin}{E22D}
318 \usv_set:nnn {stixcal}{num}{E262}
319 \usv_set:nnn {scr}{num}{48}
320 \usv_set:nnn {it}{num}{48}

321 \usv_set:nnn {stixsfitslash}{Latin}{E294}
322 \usv_set:nnn {stixsfitslash}{latin}{E2C8}
323 \usv_set:nnn {stixsfitslash}{greek}{E32C}
324 \usv_set:nnn {stixsfitslash}{epsilon}{E37A}
325 \usv_set:nnn {stixsfitslash}{vartheta}{E35E}
326 \usv_set:nnn {stixsfitslash}{varkappa}{E374}
327 \usv_set:nnn {stixsfitslash}{phi}{E360}
328 \usv_set:nnn {stixsfitslash}{varrho}{E376}
329 \usv_set:nnn {stixsfitslash}{varpi}{E362}
330 \usv_set:nnn {stixsfitslash}{digamma}{E36A}

Bold

331 \usv_set:nnn {stixbfupslash}{Greek}{E2FD}
332 \usv_set:nnn {stixbfupslash}{Digamma}{E369}
333 \usv_set:nnn {stixfbfb}{A}{E38A}
334 \usv_set:nnn {stixfbfb}{B}{E38B}
335 \usv_set:nnn {stixfbfb}{E}{E38D}
336 \usv_set:nnn {stixfbfb}{F}{E38E}
337 \usv_set:nnn {stixfbfb}{G}{E38F}
338 \usv_set:nnn {stixfbfb}{I}{E390}
339 \usv_set:nnn {stixfbfb}{J}{E391}
340 \usv_set:nnn {stixfbfb}{K}{E392}
341 \usv_set:nnn {stixfbfb}{L}{E393}
342 \usv_set:nnn {stixfbfb}{M}{E394}
343 \usv_set:nnn {stixfbfb}{O}{E395}
344 \usv_set:nnn {stixfbfb}{S}{E396}
345 \usv_set:nnn {stixfbfb}{T}{E397}
346 \usv_set:nnn {stixfbfb}{U}{E398}
347 \usv_set:nnn {stixfbfb}{V}{E399}
348 \usv_set:nnn {stixfbfb}{W}{E39A}
349 \usv_set:nnn {stixfbfb}{X}{E39B}
350 \usv_set:nnn {stixfbfb}{Y}{E39C}
351 \usv_set:nnn {stixfbfb}{a}{E39D}
352 \usv_set:nnn {stixfbfb}{b}{E39E}
353 \usv_set:nnn {stixfbfb}{c}{E39F}
354 \usv_set:nnn {stixfbfb}{f}{E3A2}
355 \usv_set:nnn {stixfbfb}{g}{E3A3}
356 \usv_set:nnn {stixfbfb}{h}{E3A4}
357 \usv_set:nnn {stixfbfb}{k}{E3A7}
358 \usv_set:nnn {stixfbfb}{l}{E3A8}
359 \usv_set:nnn {stixfbfb}{m}{E3A9}
360 \usv_set:nnn {stixfbfb}{n}{E3AA}
361 \usv_set:nnn {stixfbfb}{o}{E3AB}
362 \usv_set:nnn {stixfbfb}{p}{E3AC}
363 \usv_set:nnn {stixfbfb}{q}{E3AD}
364 \usv_set:nnn {stixfbfb}{r}{E3AE}
365 \usv_set:nnn {stixfbfb}{s}{E3AF}
366 \usv_set:nnn {stixfbfb}{t}{E3B0}
367 \usv_set:nnn {stixfbfb}{u}{E3B1}
368 \usv_set:nnn {stixfbfb}{v}{E3B2}
369 \usv_set:nnn {stixfbfb}{w}{E3B3}
370 \usv_set:nnn {stixfbfb}{x}{E3B4}
371 \usv_set:nnn {stixfbfb}{y}{E3B5}
372 \usv_set:nnn {stixfbfb}{z}{E3B6}
373 \usv_set:nnn {stixbfsfup}{Numerals}{E3B7}

Bold Italic

374 \usv_set:nnn {stixbfsfit}{Numerals}{E1F6}

375 \usv_set:nnn {stixfbbit}{A}{E200}
376 \usv_set:nnn {stixfbbit}{B}{E201}
377 \usv_set:nnn {stixfbbit}{E}{E203}
378 \usv_set:nnn {stixfbbit}{F}{E204}
379 \usv_set:nnn {stixfbbit}{G}{E205}
380 \usv_set:nnn {stixfbbit}{I}{E206}
381 \usv_set:nnn {stixfbbit}{J}{E207}
382 \usv_set:nnn {stixfbbit}{K}{E208}
383 \usv_set:nnn {stixfbbit}{L}{E209}
384 \usv_set:nnn {stixfbbit}{M}{E20A}
385 \usv_set:nnn {stixfbbit}{O}{E20B}
386 \usv_set:nnn {stixfbbit}{S}{E20C}
387 \usv_set:nnn {stixfbbit}{T}{E20D}
388 \usv_set:nnn {stixfbbit}{U}{E20E}
389 \usv_set:nnn {stixfbbit}{V}{E20F}
390 \usv_set:nnn {stixfbbit}{W}{E210}
391 \usv_set:nnn {stixfbbit}{X}{E211}
392 \usv_set:nnn {stixfbbit}{Y}{E212}

393 \usv_set:nnn {stixfbbit}{a}{E213}
394 \usv_set:nnn {stixfbbit}{b}{E214}
395 \usv_set:nnn {stixfbbit}{c}{E215}
396 \usv_set:nnn {stixfbbit}{e}{E217}
397 \usv_set:nnn {stixfbbit}{f}{E218}
398 \usv_set:nnn {stixfbbit}{g}{E219}
399 \usv_set:nnn {stixfbbit}{h}{E21A}
400 \usv_set:nnn {stixfbbit}{k}{E21D}
401 \usv_set:nnn {stixfbbit}{l}{E21E}
402 \usv_set:nnn {stixfbbit}{m}{E21F}
403 \usv_set:nnn {stixfbbit}{n}{E220}
404 \usv_set:nnn {stixfbbit}{o}{E221}
405 \usv_set:nnn {stixfbbit}{p}{E222}
406 \usv_set:nnn {stixfbbit}{q}{E223}
407 \usv_set:nnn {stixfbbit}{r}{E224}
408 \usv_set:nnn {stixfbbit}{s}{E225}
409 \usv_set:nnn {stixfbbit}{t}{E226}
410 \usv_set:nnn {stixfbbit}{u}{E227}
411 \usv_set:nnn {stixfbbit}{v}{E228}
412 \usv_set:nnn {stixfbbit}{w}{E229}
413 \usv_set:nnn {stixfbbit}{x}{E22A}
414 \usv_set:nnn {stixfbbit}{y}{E22B}
415 \usv_set:nnn {stixfbbit}{z}{E22C}

416 \usv_set:nnn {stixbfcalf}{Latin}{E247}

417 \usv_set:nnn {stixbfitslash}{Latin}{E295}
418 \usv_set:nnn {stixbfitslash}{latin}{E2C9}
419 \usv_set:nnn {stixbfitslash}{greek}{E32D}
420 \usv_set:nnn {stixsfitslash}{epsilon}{E37B}
421 \usv_set:nnn {stixsfitslash}{vartheta}{E35F}
422 \usv_set:nnn {stixsfitslash}{varkappa}{E375}

```
423 \usv_set:nnn {stixsfitslash}{phi}{"E361}  
424 \usv_set:nnn {stixsfitslash}{varrho}{"E377}  
425 \usv_set:nnn {stixsfitslash}{varpi}{"E363}  
426 \usv_set:nnn {stixsfitslash}{digamma}{"E36B}  
427 </stix>
```


File IX

um-code-setchar.dtx

10 *Setting up maths chars*

```
1 (*package)
```

10.1 *A token list to contain the data of the math table*

Instead of `\input`-ing the unicode math table every time we want to re-read its data, we save it within a macro. This has two advantages: 1. it should be slightly faster, at the expense of memory; 2. we don't need to worry about catcodes later, since they're frozen at this point.

In time, the case statement inside `set_mathsymbol` will be moved in here to avoid re-running it every time.

```
2 \group_begin:
3   \file_get:nnN {unicode-math-table.tex} {} \l_@@_mathtable_tl
4   \cs_set:Npn \UnicodeMathSymbol #1#2#3#4
5     {
6       \exp_not:n { \_@@_sym:nnn {#1} {#2} {#3} }
7     }
8   \tl_gset:Nx \g_@@_mathtable_tl {\l_@@_mathtable_tl}
9 \group_end:
```

`\@@_input_math_symbol_table`: This function simply expands to the token list containing all the data.

```
10 \@@_cs_new:Nn \@@_input_math_symbol_table: {\g_@@_mathtable_tl}
```

10.2 *Definitions of the active math characters*

Ensure catcodes are appropriate; make sure `#` is an 'other' so that we don't get confused with `\mathoctothorpe`.

```
11 \AtBeginDocument{\@@_define_math_chars:}
12 \@@_cs_new:Nn \@@_define_math_chars:
13 {
14   \group_begin:
15     \cs_set:Npn \_@@_sym:nnn ##1##2##3
16     {
17       \tl_if_in:nnT
18       { \mathord \mathalpha \mathbin \mathrel \mathpunct \mathop \mathfence }
19       {##3}
20       {
21         \exp_last_unbraced:NNx \cs_gset_eq:NN ##2 { \char_generate:nn {##1} {12} }
22       }
23     }
24   \@@_input_math_symbol_table:
25 \group_end:
26 }
```

10.3 Commands for each symbol/glyph/char

`\@@_set_mathsymbol:nNNn` #1 : A L^AT_EX symbol font, e.g., operators
 #2 : Symbol macro, e.g., `\alpha`
 #3 : Type, e.g., `\mathalpha`
 #4 : Slot, e.g., "221E

There are a bunch of tests to perform to process the various characters. The following assignments should all be fairly straightforward.

The catcode setting is to work around (strange?) behaviour in LuaTeX in which catcode 11 characters don't have italic correction for maths. We don't adjust ascii chars, however, because certain punctuation should not have their catcodes changed.

```

27 \cs_set:Nn \@@_set_mathsymbol:nNNn
28 {
29   \bool_lazy_and:nnT
30     {
31       \int_compare_p:nNn {#4} > {127}
32     }
33     {
34       \int_compare_p:nNn { \char_value_catcode:n {#4} } = {11}
35     }
36     { \char_set_catcode_other:n {#4} }
37
38   \token_case_meaning:Nn #3
39   {
40     \mathord { \@@_set_mathcode:nnn {#4} {#3} {#1} }
41     \mathalpha { \@@_set_mathcode:nnn {#4} {#3} {#1} }
42     \mathbin { \@@_set_mathcode:nnn {#4} {#3} {#1} }
43     \mathrel { \@@_set_mathcode:nnn {#4} {#3} {#1} }
44     \mathpunct { \@@_set_mathcode:nnn {#4} {#3} {#1} }
45     \mathop { \@@_set_big_operator:nnn {#1} {#2} {#4} }
46     \mathopen { \@@_set_math_open:nnn {#1} {#2} {#4} }
47     \mathclose { \@@_set_math_close:nnn {#1} {#2} {#4} }
48     \mathfence { \@@_set_math_fence:nnnn {#1} {#2} {#3} {#4} }
49     \mathaccent
50     { \@@_set_math_accent:Nnnn #2 {fixed} {#1} {#4} }
51     \mathbotaccent
52     { \@@_set_math_accent:Nnnn #2 {bottom~ fixed} {#1} {#4} }
53     \mathaccentwide
54     { \@@_set_math_accent:Nnnn #2 {} {#1} {#4} }
55     \mathbotaccentwide
56     { \@@_set_math_accent:Nnnn #2 {bottom} {#1} {#4} }
57     \mathover
58     { \@@_set_math_overunder:Nnnn #2 {} {#1} {#4} }
59     \mathunder
60     { \@@_set_math_overunder:Nnnn #2 {bottom} {#1} {#4} }
61     \mathaccentoverlay
62 <LU> { \@@_set_math_accent:Nnnn #2 {overlay~ fixed} {#1} {#4} }

```

```

63 (XE)      { \@@_set_math_accent:Nnnn #2 {} {#1} {#4} }
64      }
65      }

66 \edef\mathfence{\string\mathfence}
67 \edef\mathover{\string\mathover}
68 \edef\mathunder{\string\mathunder}
69 \edef\mathbotaccent{\string\mathbotaccent}
70 \edef\mathaccentwide{\string\mathaccentwide}
71 \edef\mathaccentoverlay{\string\mathaccentoverlay}
72 \edef\mathbotaccentwide{\string\mathbotaccentwide}

```

`\@@_set_big_operator:nnn` #1 : Symbol font name
#2 : Macro to assign
#3 : Glyph slot

In the examples following, say we're defining for the symbol `\sum` (Σ). 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. This involves three steps:

- The active math char is defined to expand to the macro `\sum_sym`. (Later, the control sequence `\sum` will be assigned the math char.)
- Declare the plain old `mathchardef` for the control sequence `\sumop`. (This follows the convention of \LaTeX /amsmath.)
- Define `\sum_sym` as `\sumop`, followed by `\nolimits` if necessary.

Whether the `\nolimits` suffix is inserted is controlled by the token list `\l_@@_nolimits_tl`, which contains a list of such characters. This list is checked dynamically to allow it to be updated mid-document.

Examples of expansion, by default, for two big operators:

```

( \sum → )  $\Sigma$  → \sum_sym → \sumop\nolimits
( \int → )  $\int$  → \int_sym → \intop

```

```

73 \cs_new:Nn \@@_set_big_operator:nnn
74 {
75   \@@_char_gmake_mathactive:n {#3}
76   \cs_set_protected_nopar:Npx \@@_tmpa: { \exp_not:c { \cs_to_str:N #2 _sym } }
77   \char_gset_active_eq:nN {#3} \@@_tmpa:
78
79   \@@_set_mathchar:cNnn { \cs_to_str:N #2 op } \mathop {#1} {#3}
80
81   \cs_gset:cpx { \cs_to_str:N #2 _sym }
82   {
83     \exp_not:c { \cs_to_str:N #2 op }
84     \exp_not:n { \tl_if_in:NnT \l_@@_nolimits_tl {#2} \nolimits }
85   }
86 }

```

```

\@@_set_math_open:nnn #1 : Symbol font name
#2 : Macro to assign
#3 : Glyph slot
87 \cs_new:Nn \@@_set_math_open:nnn
88 {
89   \tl_if_in:NnTF \l_@@_radicals_tl {#2}
90   {
91     \cs_if_exist:NF #2
92     {
93       %% todo: check if the check is necessary
94       \cs_gset_protected_nopar:Npx #2 { \exp_not:c { \cs_to_str:N #2 sign } }
95     }
96     \cs_gset_protected_nopar:cpx { \cs_to_str:N #2 sign }
97     {
98       \@@_radical:nn {#1} {#3}
99     }
100    \tl_if_exist:cF {c_@@_radical_\cs_to_str:N #2_tl}
101    {
102      \tl_const:cn {c_@@_radical_\cs_to_str:N #2_tl} {\use:c{sym #1}~ #3}
103    }
104  }
105  {
106    \@@_set_delcode:nnn {#1} {#3} {#3}
107    \@@_set_mathcode:nnn {#3} \mathopen {#1}
108    \cs_gset_protected_nopar:Npx #2
109    { \@@_delimiter:Nnn \mathopen {#1} {#3} }
110  }
111 }

```

```

\@@_set_math_close:nnn #1 : Symbol font name
#2 : Macro to assign
#3 : Glyph slot
112 \cs_new:Nn \@@_set_math_close:nnn
113 {
114   \@@_set_delcode:nnn {#1} {#3} {#3}
115   \@@_set_mathcode:nnn {#3} \mathclose {#1}
116   \cs_gset_protected_nopar:Npx #2
117   { \@@_delimiter:Nnn \mathclose {#1} {#3} }
118 }

```

```

\@@_set_math_fence:nnnn #1 : Symbol font name
#2 : Macro to assign
#3 : Type, e.g., \mathalpha
#4 : Glyph slot
119 \cs_new:Nn \@@_set_math_fence:nnnn
120 {
121   \@@_set_mathcode:nnn {#4} {#3} {#1}
122   \@@_set_delcode:nnn {#1} {#4} {#4}

```

```

123 \cs_gset_protected_nopar:cpx {l \cs_to_str:N #2}
124 { \@@_delimiter:Nnn \mathopen {#1} {#4} }
125 \cs_gset_protected_nopar:cpx {r \cs_to_str:N #2}
126 { \@@_delimiter:Nnn \mathclose {#1} {#4} }
127 }

```

`\@@_set_math_accent:Nnnn` #1 : Accend command
#2 : Accent type (string)
#3 : Symbol font name
#4 : Glyph slot

```

128 \cs_new:Nn \@@_set_math_accent:Nnnn
129 {
130 \cs_gset_protected_nopar:Npx #1
131 { \@@_accent:nnn {#2} {#3} {#4} }
132 }

```

`\@@_set_math_overunder:Nnnn` #1 : Accend command
#2 : Accent type (string)
#3 : Symbol font name
#4 : Glyph slot

```

133 \cs_new:Nn \@@_set_math_overunder:Nnnn
134 {
135 \cs_gset_protected_nopar:Npx #1 ##1
136 {
137 \mathop
138 { \@@_accent:nnn {#2} {#3} {#4} {#1} }
139 % TODO: remove braces above ^^ which work around a LuaTeX bug
140 \limits
141 }
142 }

```

```

143 </package>

```

File X

um-code-mathtext.dtx

11 Maths text commands

1 *(*package)*

11.1 `\setmathfontface`

`\@@_setmathfontface:Nnn` Interface around `\SetMathAlphabet`.

```
2 \keys_define:nn {@@_mathface}
3 {
4   version .tl_set:N = \l_@@_mversion_tl
5 }
6 \@@_cs_new:Nn \@@_setmathfontface:Nnn
7 {
8   \tl_clear:N \l_@@_mversion_tl
9
10  \keys_set_known:nnN {@@_mathface} {#2} \l_@@_keyval_clist
11
12  \fontspec_set_family:Nxx \l_@@_tmpa_tl
13  { ItalicFont={}, BoldFont={}, SmallCapsFont={}, \exp_not:V \l_@@_keyval_clist }
14  { #3 }
15
16  \tl_if_empty:NT \l_@@_mversion_tl
17  {
18    \tl_set:Nn \l_@@_mversion_tl {normal}
19    \DeclareMathAlphabet #1 {\g_fontspec_encoding_tl} {\l_@@_tmpa_tl} {\md-
20    default} {\shapedefault}
21  }
22  \SetMathAlphabet #1 {\l_@@_mversion_tl} {\g_fontspec_encoding_tl} {\l_@@_tmpa_tl} {\md-
23  default} {\shapedefault}
24
25  % integrate with fontspec's \setmathrm etc:
26  \token_case_meaning:Nn #1
27  {
28    \mathrm { \cs_gset_eq:NN \g__fontspec_mathrm_tl \l_@@_tmpa_tl }
29    \mathsf { \cs_gset_eq:NN \g__fontspec_mathsf_tl \l_@@_tmpa_tl }
30    \mathtt { \cs_gset_eq:NN \g__fontspec_mathtt_tl \l_@@_tmpa_tl }
31  }
```

11.2 Hooks into $\LaTeX 2_{\epsilon}$

Switching to a different style of alphabetic symbols was traditionally performed with commands like `\mathbf`, which literally changes fonts to access alternate

symbols. This is not as simple with Unicode fonts.

In traditional T_EX maths font setups, you simply switch between different ‘families’ (`\fam`), which is analogous to changing from one font to another—a symbol such as ‘a’ will be upright in one font, bold in another, and so on. In `pkgunicode-math`, a different mechanism is used to switch between styles. For every letter (start with `ascii a-zA-Z` and numbers to keep things simple for now), they are assigned a ‘mathcode’ with `\Umathcode` that maps from input letter to output font glyph slot. This is done with the equivalent of

```
% \Umathcode`a = 7 1 "1D44E\relax
% \Umathcode`b = 7 1 "1D44F\relax
% \Umathcode`c = 7 1 "1D450\relax
% ...
```

When switching from regular letters to, say, `\mathrm`, we now need to execute a new mapping:

```
% \Umathcode`a = 7 1 `a\relax
% \Umathcode`b = 7 1 `b\relax
% \Umathcode`c = 7 1 `c\relax
% ...
```

This is fairly straightforward to perform when we’re defining our own commands such as `\symbolf` and so on. However, this means that ‘classical’ T_EX font setups will break, because with the original mapping still in place, the engine will be attempting to insert unicode maths glyphs from a standard font.

`\use@mathgroup` To overcome this, we patch `\use@mathgroup`, which is only used inside of commands such as `\mathXYZ`, so this shouldn’t have any side-effects. Omit the test for math mode because this is only called *inside* `\mathrm` or similar, which already has a math mode check.

```
32 \cs_set:Npn \use@mathgroup #1 #2
33 {
34   \math@bgroup
35   \cs_if_eq:cNF {M@f@encoding} #1 {#1}
36   \@_switch_to:n {literal}
37   \@_mathgroup_set:n {#2}
38   \math@egroup
39 }
```

In LaTeX maths, the command `\operator@font` is defined that switches to the operator `mathgroup`. The classic example is the `\sin` in `sin{x}`; essentially we’re using `\mathrm` to typeset the upright symbols, but the syntax is `{\operator@font sin}`. I thought that hooking into `\operator@font` would be hard because all other maths font selection in 2e uses `\mathrm{...}` style. Then reading `source2e` a little more I stumbled upon `\@fontswitch`. Reimplement that here to avoid `\bgroup/\egroup`.

`\operator@font`

```

40 \cs_set_protected:Npn \operator@font
41 {
42   \@@_switch_to:n {literal}
43   \@@_fontswitch:n { \g_@@_operator_mathfont_tl }
44 }

```

`\@@_fontswitch:n` Omit the check for math mode as #1 should do that for us.

```

45 \cs_set:Nn \@@_fontswitch:n
46 {
47   \cs_set_eq:NN \math@bgroup \scan_stop:
48   \cs_set_eq:NN \@@_group_begin: \scan_stop:
49   \cs_set:Npn \@@_group_end:n % takes no argument in this case
50   {
51     \cs_set_eq:NN \@@_group_begin: \@@_group_begin_frozen:
52     \cs_set_eq:NN \@@_group_end:n \@@_group_end_frozen:n
53     \cs_set_eq:NN \math@bgroup \@@@math@bgroup
54     \cs_set_eq:NN \math@egroup \@@@math@egroup
55   }
56   \cs_set_eq:NN \math@egroup \@@_group_end:n
57   #1 \scan_stop:
58 }

```

11.3 Hooks into fontspec

Historically, `\mathrm` and so on were completely overwritten by `unicode-math`, and `fontspec`'s methods for setting these fonts in the classical manner were bypassed.

While we could now re-activate the way that `fontspec` does the following, because we can now change maths fonts whenever it's better to define new commands in `unicode-math` to define the `\mathXYZ` fonts.

11.3.1 Text font

```

59 \cs_generate_variant:Nn \tl_if_eq:nnT {o}
60 \@@_cs_set:Nn \__fontspec_setmainfont_hook:nn
61 {
62   \tl_if_eq:onT {\g__fontspec_mathrm_tl} {\rmdefault}
63   {
64     (XE) \fontspec_gset_family:Nnn \g__fontspec_mathrm_tl {#1} {#2}
65     (LU) \fontspec_gset_family:Nnn \g__fontspec_mathrm_tl {Renderer=Basic,#1} {#2}
66     \__fontspec_setmathrm_hook:nn {#1} {#2}
67   }
68 }
69 \@@_cs_set:Nn \__fontspec_setsansfont_hook:nn
70 {
71   \tl_if_eq:onT {\g__fontspec_mathsf_tl} {\sfdefault}
72   {
73     (XE) \fontspec_gset_family:Nnn \g__fontspec_mathsf_tl {#1} {#2}
74     (LU) \fontspec_gset_family:Nnn \g__fontspec_mathsf_tl {Renderer=Basic,#1} {#2}
75     \__fontspec_setmathsf_hook:nn {#1} {#2}

```



```

76     }
77   }
78 \@@_cs_set:Nn \__fontspec_setmonofont_hook:nn
79   {
80     \tl_if_eq:onT {\g__fontspec_mathtt_tl} {\ttdefault}
81     {
82 (XE) \fontspec_gset_family:Nnn \g__fontspec_mathtt_tl {#1} {#2}
83 (LU) \fontspec_gset_family:Nnn \g__fontspec_mathtt_tl {Renderer=Basic,#1} {#2}
84     \__fontspec_setmathtt_hook:nn {#1} {#2}
85   }
86 }

```

11.3.2 Maths font

If the maths fonts are set explicitly, then the text commands above will not execute their branches to set the maths font alphabets.

Helper macro for looking up customisable series' by family (new L^AT_EX_{2_ε} feature 2020).

```

87 \cs_new:Nn \@@_rm_series_default:n
88   {
89     \ifcsname #1series@rm\endcsname
90     \csname #1series@rm\endcsname
91     \else
92     \csname #1default\endcsname
93     \fi
94   }
95 \@@_cs_set:Nn \__fontspec_setmathrm_hook:nn
96   {
97     \SetMathAlphabet\mathrm{normal}\g_fontspec_encoding_tl\g__fontspec_mathrm_tl{\@@_rm_series_de
98     \SetMathAlphabet\mathit{normal}\g_fontspec_encoding_tl\g__fontspec_mathrm_tl{\@@_rm_series_de
99     \SetMathAlphabet\mathbf{normal}\g_fontspec_encoding_tl\g__fontspec_mathrm_tl{\@@_rm_series_de
100   }
101 \@@_cs_set:Nn \__fontspec_setboldmathrm_hook:nn
102   {
103     \SetMathAlphabet\mathrm{bold}\g_fontspec_encoding_tl\g__fontspec_bfmathrm_tl{\@@_rm_series_de
104     \SetMathAlphabet\mathbf{bold}\g_fontspec_encoding_tl\g__fontspec_bfmathrm_tl{\@@_rm_series_de
105     \SetMathAlphabet\mathit{bold}\g_fontspec_encoding_tl\g__fontspec_bfmathrm_tl\mddefault\itdefa
106   }
107 \@@_cs_set:Nn \__fontspec_setmathsf_hook:nn
108   {
109     \SetMathAlphabet\mathsf{normal}\g_fontspec_encoding_tl\g__fontspec_mathsf_tl{\@@_rm_series_de
110     \SetMathAlphabet\mathsf{bold} \g_fontspec_encoding_tl\g__fontspec_mathsf_tl{\@@_rm_series_de
111   }
112 \@@_cs_set:Nn \__fontspec_setmathtt_hook:nn
113   {
114     \SetMathAlphabet\mathtt{normal}\g_fontspec_encoding_tl\g__fontspec_mathtt_tl{\@@_rm_series_de
115     \SetMathAlphabet\mathtt{bold} \g_fontspec_encoding_tl\g__fontspec_mathtt_tl{\@@_rm_series_de
116   }

```

I can't quite remember the logic behind the following two.

If fontspec has been loaded and `\setmathsf` (etc) run, this syncs things up:

```
117 \tl_if_eq:onF {\g__fontspec_mathrm_tl} {\rmdefault} { \__fontspec_setmathrm_hook:nn {} {} }
118 \tl_if_eq:onF {\g__fontspec_mathsf_tl} {\sfdefault} { \__fontspec_setmathsf_hook:nn {} {} }
119 \tl_if_eq:onF {\g__fontspec_mathtt_tl} {\ttdefault} { \__fontspec_setmathtt_hook:nn {} {} }
```

I suppose this is to make things work if neither fontspec or unicode-math load any fonts: (I should check that)

```
120 \AtBeginDocument
121 {
122   \tl_if_eq:onT {\g__fontspec_mathrm_tl} {\rmdefault} { \__fontspec_setmathrm_hook:nn {} {} }
123   \tl_if_eq:onT {\g__fontspec_mathsf_tl} {\sfdefault} { \__fontspec_setmathsf_hook:nn {} {} }
124   \tl_if_eq:onT {\g__fontspec_mathtt_tl} {\ttdefault} { \__fontspec_setmathtt_hook:nn {} {} }
125 }
126 </package>
```

File XI

um-code-main.dtx

12 The main `\setmathfont` macro

```
1 (*package)
@@_setmathfont:nn #1 : keyval options
#2 : font name/file
2 \@@_cs_new:Nn \@@_setmathfont:nn
3 {
    • Initialise all local variables.
    • Erase any conception LATEX has of previously defined math symbol fonts;
      this allows \DeclareSymbolFont at any point in the document.
    • Grab the current size information: (is this robust enough? Maybe it should
      be preceded by \normalsize). 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.
4
5 \@@_init:n {#2}
6 \cs_set_eq:NN \glb@currsizsize \scan_stop:
7 \cs_if_exist:cF { S@ \f@size } { \calculate@math@sizes }
8 \use:c { S@ \f@size }
9
10 \keys_set_known:nnN {unicode-math} {#1} \l_@@_unknown_keys_clist
11
12 \bool_if:NT \g_@@_init_bool \@@_fontspec_trial_font:
13 \bool_if:NT \g_@@_init_bool \@@_declare_math_sizes:
14
15 \@@_fontspec_select_font:
16 \@@_setup_math_fam:
17 \bool_if:NT \g_@@_init_bool
18 {
19 \@@_setup_legacy_fam_two:
20 \@@_setup_legacy_fam_three:
21 }
22
23 \@@_input_math_symbol_table:
24
```

- the ‘once-off’ setup that doesn’t need to be per-font
- remap symbols that don’t take their natural mathcode;
- activate any symbols that need to be math-active;

- assign delimiter codes for symbols that need to grow;
- setup the maths alphabets (`\symbf` etc.) — this is an extensive part of the code; see Section 15;

```

25 \bool_if:NT \g_@@_init_bool \@@_onceoff_setup:
26 \@@_remap_symbols:
27 \@@_setup_mathactives:
28 \@@_setup_delcodes:
29 \@@_setup_alphabets:
30
31 %% TODO: what of the above should only be run for the "de-
    fault"/"main" font?
32
33 \bool_if:NTF \g_@@_init_bool
34 {
35   \bool_gset_true:N \g_@@_main_font_defined_bool
36 (LU) \@@_mathparam_store:
37   \@@_log:n {default-math-font}
38 }
39 {
40 (LU) \@@_mathparam_restore:
41 }
42 }

```

Fall-back font Want to load Latin Modern Math if nothing else. This needs to happen early so that all of the font-loading machinery executes before the other ‘At-BeginDocument’ code.

```

43 \AtBeginDocument { \bool_if:NF \g_@@_main_font_defined_bool \@@_load_lm: }
44 \@@_cs_new:Nn \@@_load_lm:
45 {
46   \setmathfont{latinmodern-math.otf}[BoldFont={latinmodern-math.otf}]
47 }

```

`\@@_init:n` Reset local variables. Default to defining the font for every math symbol character.

```

48 \@@_cs_new:Nn \@@_init:n
49 {
50   \tl_set:Nn \l_@@_fontname_tl {#1}
51   \bool_gset_true:N \g_@@_ot_math_bool
52   \tl_set:Nn \l_@@_mversion_tl {normal}
53   \tl_set:Nn \l_@@_symfont_label_tl {operators}
54
55   \tl_set:Nn \l_@@_script_features_tl {Style=MathScript}
56   \tl_set:Nn \l_@@_sscript_features_tl {Style=MathScriptScript}
57   \tl_set_eq:NN \l_@@_script_font_tl \l_@@_fontname_tl
58   \tl_set_eq:NN \l_@@_sscript_font_tl \l_@@_fontname_tl
59
60   \bool_gset_true:N \g_@@_init_bool
61   \seq_gclear:N \g_@@_char_range_seq

```

```

62 \clist_clear:N \l_@@_mathmap_charints_clist
63 \seq_gclear:N \g_@@_mathalph_seq
64 \seq_clear:N \l_@@_missing_alph_seq
65
66 \cs_set_eq:NN \@@_sym:nnn \@@_process_symbol_noparse:nnn
67 \cs_set_eq:NN \@@_remap_symbol:nnn \@@_remap_symbol_noparse:nnn
68 \cs_set_eq:NN \@@_maybe_init_alphabet:n \@@_init_alphabet:n
69 \cs_set_eq:NN \@@_assign_delcode:nn \@@_assign_delcode_noparse:nn
70 \cs_set_eq:NN \@@_make_mathactive:nNN \@@_make_mathactive_noparse:nNN
71 }

```

`\@@_declare_math_sizes:` Set the math sizes according to the recommended font parameters.

```

72 \tl_new:N \g_@@_main_font_cmd_tl
73 \cs_new:Nn \@@_sf_size: { \@@_fontdimen_pc_to_pt:nN {10} \g_@@_trial_font }
74 \cs_new:Nn \@@_ssf_size: { \@@_fontdimen_pc_to_pt:nN {11} \g_@@_trial_font }
75 \@@_cs_new:Nn \@@_declare_math_sizes:
76 {
77   \fp_gset:Nn \g_@@_size_tfsf_fp { (\f@size + \@@_sf_size: )/2 }
78   \fp_gset:Nn \g_@@_size_sfssf_fp { (\@@_sf_size: + \@@_ssf_size:)/2 }
79
80   \dim_compare:nF { \fontdimen 10 \g_@@_trial_font == 0pt }
81   {
82     \DeclareMathSizes { \f@size } { \f@size } { \@@_sf_size: } { \@@_ssf_size: }
83   }
84 }

```

`\@@_fontspec_trial_font:`

```

85 \@@_cs_new:Nn \@@_fontspec_trial_font:
86 {
87   \tl_set:Nx \l_@@_font_keyval_tl
88   {
89     (LU) Renderer = Basic,
90     BoldItalicFont = {}, ItalicFont = {}, SmallCapsFont = {},
91     Script = Math,
92     (LU) RawFeature = {mathfontdimen=xetex},
93     \l_@@_unknown_keys_clist
94   }
95
96   \fontspec_set_family:Nxn \l_@@_trial_family_tl { \l_@@_font_keyval_tl } { \l_@@_fontname_tl }
97
98   \group_begin:
99     \fontfamily { \l_@@_trial_family_tl } \selectfont
100   \exp_last_unbraced:NNo \@@_fontface_gset_eq:NN \g_@@_trial_font \font@name
101   \fontspec_if_script:nF {math}
102   {
103     \@@_warning:n {not-ot-math}
104     \bool_gset_false:N \g_@@_ot_math_bool
105     \bool_gset_false:N \g_@@_init_bool
106   }

```

```

107   \group_end:
108
109   }

\@@_fontspec_select_font:
110 \@@_cs_new:Nn \@@_fontspec_select_font:
111   {
112     \tl_set:Nx \l_@@_font_keyval_tl
113       {
114 (LU)   Renderer = Basic,
115         BoldItalicFont = {}, ItalicFont = {}, SmallCapsFont = {},
116         Script = Math,
117         SizeFeatures =
118           {
119             {
120               Size = \fp_use:N \g_@@_size_tfsf_fp -
121             } ,
122             {
123               Size = \fp_use:N \g_@@_size_sfssf_fp - \fp_use:N \g_@@_size_tfsf_fp ,
124               Font = \l_@@_script_font_tl ,
125               \l_@@_script_features_tl
126             } ,
127             {
128               Size = - \fp_use:N \g_@@_size_sfssf_fp ,
129               Font = \l_@@_sscript_font_tl ,
130               \l_@@_sscript_features_tl
131             }
132           } ,
133 (LU)   RawFeature = {mathfontdimen=xetex},
134         \l_@@_unknown_keys_clist
135       }
136
137     \fontspec_set_family:Nxn \l_@@_family_tl {\l_@@_font_keyval_tl} {\l_@@_fontname_tl}
138
139     \int_gincr:N \g_@@_fonts_used_int
140     \group_begin:
141       \fontfamily { \l_@@_family_tl } \selectfont
142       \exp_last_unbraced:Nno \@@_fontface_gset_eq:cN {g_@@_mathfont_ \int_use:N \g_@@_fonts_used_int}
143       \tl_gset:Nx \g_@@_curr_font_cmd_tl { \exp_not:c {g_@@_mathfont_ \int_use:N \g_@@_fonts_used_int} }
144       \bool_if:NT \g_@@_init_bool
145     {
146       \exp_last_unbraced:Nno \@@_fontface_gset_eq:NN \l_@@_font \font@name
147     }
148     \group_end:
149   }

150 \tl_gset:Nn \g_@@_main_font_cmd_tl { \l_@@_font }
151 \tl_gset:Nn \g_@@_sqrt_font_cmd_tl { \l_@@_font }
152 \tl_gset:Nn \g_@@_prime_font_cmd_tl { \l_@@_font }

```

`\@@_setup_math_fam:`

```
153 \@@_cs_new:Nn \@@_setup_math_fam:
154 {
155   \cs_if_exist:cF { sym \l_@@_symfont_label_tl }
156   {
157     \DeclareSymbolFont{\l_@@_symfont_label_tl}
158     {\encodingdefault}{\l_@@_family_tl}{\mddefault}{\shapedefault}
159   }
160   \SetSymbolFont{\l_@@_symfont_label_tl}{\l_@@_mversion_tl}
161   {\encodingdefault}{\l_@@_family_tl}{\mddefault}{\shapedefault}
```

Set the bold math version.

```
162   \str_if_eq:eeT {\l_@@_mversion_tl} {normal}
163   {
164     \SetSymbolFont{\l_@@_symfont_label_tl}{bold}
165     {\encodingdefault}{\l_@@_family_tl}{\bfdefault}{\shapedefault}
166   }
167 }
```

`\@@_setup_legacy_fam_two:` \TeX won't load the same font twice at the same scale, so we need to magnify this one by an imperceptable amount. Note that for extreme font sizes, this scaling value might need to be adjusted. 1.0001 should be enough for reasonable use cases however.

```
168 \@@_cs_new:Nn \@@_setup_legacy_fam_two:
169 {
170   \fontspec_set_family:Nxn \l_@@_fam_two_tl
171   {
172     \l_@@_font_keyval_tl,
173     (LU) RawFeature = {mathfontdimen=tex2},
174     (*XE)
175     ScaleAgain = 1.0001,
176     FontAdjustment =
177     {
178       \@@_copy_fontdimen:nnN { 8} {43} \g_@@_main_font_cmd_tl
179       \@@_copy_fontdimen:nnN { 9} {42} \g_@@_main_font_cmd_tl
180       \@@_copy_fontdimen:nnN {10} {32} \g_@@_main_font_cmd_tl
181       \@@_copy_fontdimen:nnN {11} {45} \g_@@_main_font_cmd_tl
182       \@@_copy_fontdimen:nnN {12} {44} \g_@@_main_font_cmd_tl
183       \@@_copy_fontdimen:nnN {13} {21} \g_@@_main_font_cmd_tl
184       \@@_copy_fontdimen:nnN {14} {21} \g_@@_main_font_cmd_tl
185       \@@_copy_fontdimen:nnN {15} {22} \g_@@_main_font_cmd_tl
186       \@@_copy_fontdimen:nnN {16} {18} \g_@@_main_font_cmd_tl
187       \@@_copy_fontdimen:nnN {17} {18} \g_@@_main_font_cmd_tl
188       \@@_copy_fontdimen:nnN {18} {24} \g_@@_main_font_cmd_tl
189       \@@_copy_fontdimen:nnN {19} {20} \g_@@_main_font_cmd_tl
190       \@@_copy_fontdimen:nnN {22} {15} \g_@@_main_font_cmd_tl
191       \@@_zero_fontdimen:n {20} % delim1 = FractionDelimiterDisplaySize
192       \@@_zero_fontdimen:n {21} % delim2 = FractionDelimiterSize
193     }
194 }
```

```

194 </XE>
195     } {\l_@@_fontname_tl}
196
197     \SetSymbolFont{symbols}{\l_@@_mversion_tl}
198     {\encodingdefault}{\l_@@_fam_two_tl}{\mddefault}{\shapedefault}
199
200     \str_if_eq:eeT {\l_@@_mversion_tl} {normal}
201     {
202         \SetSymbolFont{symbols}{bold}
203         {\encodingdefault}{\l_@@_fam_two_tl}{\bfdefault}{\shapedefault}
204     }
205 }

```

\@@_setup_legacy_fam_three: Similarly, this font is shrunk by an imperceptible amount for T_EX to load it again.

```

206 \@@_cs_new:Nn \@@_setup_legacy_fam_three:
207 {
208     \fontspec_set_family:Nxn \l_@@_fam_three_tl
209     {
210         \l_@@_font_keyval_tl,
211 <LU> RawFeature = {mathfontdimen=tex3},
212 <*XE>
213         ScaleAgain = 0.9999,
214         FontAdjustment = {
215             \@@_copy_fontdimen:nnN { 8} {48} \g_@@_main_font_cmd_tl
216             \@@_copy_fontdimen:nnN { 9} {28} \g_@@_main_font_cmd_tl
217             \@@_copy_fontdimen:nnN {10} {30} \g_@@_main_font_cmd_tl
218             \@@_copy_fontdimen:nnN {11} {29} \g_@@_main_font_cmd_tl
219             \@@_copy_fontdimen:nnN {12} {31} \g_@@_main_font_cmd_tl
220             \@@_zero_fontdimen:n {13}
221         }
222 </XE>
223     } {\l_@@_fontname_tl}
224
225     \SetSymbolFont{largesymbols}{\l_@@_mversion_tl}
226     {\encodingdefault}{\l_@@_fam_three_tl}{\mddefault}{\shapedefault}
227
228     \str_if_eq:eeT {\l_@@_mversion_tl} {normal}
229     {
230         \SetSymbolFont{largesymbols}{bold}
231         {\encodingdefault}{\l_@@_fam_three_tl}{\bfdefault}{\shapedefault}
232     }
233 }

```

\@@_onceoff_setup:

```

234 \@@_cs_new:Nn \@@_onceoff_setup:
235 {
236     \@@_set_delcode:nnn {operators} {\.\.} {0}
237 }

```


12.1 Functions for setting up symbols with mathcodes

`\@@_process_symbol_noparse:nnn` If the range font feature has been used, then only a subset of the Unicode glyphs are to be defined. See section §13.3 for the code that enables this.

```

238 \cs_set:Nn \@@_process_symbol_noparse:nnn
239   {
240     \@@_set_mathsymbol:nNn {\l_@@_symfont_label_t1} #2 #3 {#1}
241   }
242 \cs_set:Nn \@@_process_symbol_parse:nnn
243   {
244     \@@_if_char_spec:nNT {#1} {#3}
245     {
246       \@@_process_symbol_noparse:nnn {#1} {#2} {#3}
247     }
248   }

```

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

```

249 \@@_cs_new:Nn \@@_remap_symbols:
250   {
251     \@@_remap_symbol:nnn {\-} {\mathbin} {"2212}
252     \@@_remap_symbol:nnn {\*} {\mathbin} {"02217}% text asterisk to "cen-
253     tred asterisk"
254     \bool_if:NF \g_@@_literal_colon_bool
255     {
256       \@@_remap_symbol:nnn {\:} {\mathrel} {"02236}% colon to ra-
257       tio (i.e., punct to rel)
258     }
259   }

```

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

```

258 \cs_new:Nn \@@_remap_symbol_parse:nnn
259   {
260     \@@_if_char_spec:nNT {#3} {#2}
261     { \@@_remap_symbol_noparse:nnn {#1} {#2} {#3} }
262   }
263 \cs_new:Nn \@@_remap_symbol_noparse:nnn
264   {
265     \clist_map_inline:nn {#1}
266     { \@@_set_mathcode:nnnn {##1} {#2} {\l_@@_symfont_label_t1} {#3} }
267   }

```

12.2 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.

`\@@_setup_mathactives:` TODO: if not an OpenType math font, we should ignore doing anything with primes. This needs a revamped ‘range’ feature, I think.

```

268 \@@_cs_new:Nn \@@_setup_mathactives:
269   {
270   \@@_make_mathactive:nNN {"2032} \@@_prime_single_mchar \mathord
271   \@@_make_mathactive:nNN {"2033} \@@_prime_double_mchar \mathord
272   \@@_make_mathactive:nNN {"2034} \@@_prime_triple_mchar \mathord
273   \@@_make_mathactive:nNN {"2057} \@@_prime_quad_mchar \mathord
274   \@@_make_mathactive:nNN {"2035} \@@_backprime_single_mchar \mathord
275   \@@_make_mathactive:nNN {"2036} \@@_backprime_double_mchar \mathord
276   \@@_make_mathactive:nNN {"2037} \@@_backprime_triple_mchar \mathord
277   \@@_make_mathactive:nNN {\`'} \mathstraightquote \mathord
278   \@@_make_mathactive:nNN {\`\`} \mathbacktick \mathord
279   }

```

`\@@_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!

```

280 \cs_new:Nn \@@_make_mathactive_parse:nNN
281   {
282   \@@_if_char_spec:nNT {#1} #3
283     { \@@_make_mathactive_noparse:nNN {#1} #2 #3 }
284   }
285 \cs_new:Nn \@@_make_mathactive_noparse:nNN
286   {
287   \@@_set_mathchar:NNnn #2 #3 {\l_@@_symfont_label_tl} {#1}
288   \@@_char_gmake_mathactive:n {#1}
289   }

```

12.3 *Delimiter codes*

`\@@_assign_delcode:nn`

```

290 \cs_new:Nn \@@_assign_delcode_noparse:nn
291   {
292   \@@_set_delcode:nnn \l_@@_symfont_label_tl {#1} {#2}
293   }
294 \cs_new:Nn \@@_assign_delcode_parse:nn
295   {
296   \@@_if_char_spec:nNT {#2} \@nil
297     {
298     \@@_assign_delcode_noparse:nn {#1} {#2}
299     }
300   }

```

`\@@_assign_delcode:n` Shorthand.

```

301 \cs_new:Nn \@@_assign_delcode:n { \@@_assign_delcode:nn {#1} {#1} }

```

`\@@_setup_delcodes:` Some symbols that aren’t mathopen/mathclose still need to have delimiter codes assigned. The list of vertical arrows may be incomplete. On the other hand, many

fonts won't support them all being stretchy. And some of them are probably not meant to stretch, either. But adding them here doesn't hurt.

```

302 \@@_cs_new:Nn \@@_setup_delcodes:
303   {
304     \@@_assign_delcode:nn {\`\/} {\g_@@_slash_delimiter_usv}
305     \@@_assign_delcode:nn {"2044} {\g_@@_slash_delimiter_usv} % fracslash
306     \@@_assign_delcode:nn {"2215} {\g_@@_slash_delimiter_usv} % divslash
307     \@@_assign_delcode:n {"005C} % backslash
308     \@@_assign_delcode:nn {\`\<} {"27E8} % angle brackets with ascii notation
309     \@@_assign_delcode:nn {\`\>} {"27E9} % angle brackets with ascii notation
310     \@@_assign_delcode:n {"2191} % up arrow
311     \@@_assign_delcode:n {"2193} % down arrow
312     \@@_assign_delcode:n {"2195} % updown arrow
313     \@@_assign_delcode:n {"219F} % up arrow twohead
314     \@@_assign_delcode:n {"21A1} % down arrow twohead
315     \@@_assign_delcode:n {"21A5} % up arrow from bar
316     \@@_assign_delcode:n {"21A7} % down arrow from bar
317     \@@_assign_delcode:n {"21A8} % updown arrow from bar
318     \@@_assign_delcode:n {"21BE} % up harpoon right
319     \@@_assign_delcode:n {"21BF} % up harpoon left
320     \@@_assign_delcode:n {"21C2} % down harpoon right
321     \@@_assign_delcode:n {"21C3} % down harpoon left
322     \@@_assign_delcode:n {"21C5} % arrows up down
323     \@@_assign_delcode:n {"21F5} % arrows down up
324     \@@_assign_delcode:n {"21C8} % arrows up up
325     \@@_assign_delcode:n {"21CA} % arrows down down
326     \@@_assign_delcode:n {"21D1} % double up arrow
327     \@@_assign_delcode:n {"21D3} % double down arrow
328     \@@_assign_delcode:n {"21D5} % double updown arrow
329     \@@_assign_delcode:n {"21DE} % up arrow double stroke
330     \@@_assign_delcode:n {"21DF} % down arrow double stroke
331     \@@_assign_delcode:n {"21E1} % up arrow dashed
332     \@@_assign_delcode:n {"21E3} % down arrow dashed
333     \@@_assign_delcode:n {"21E7} % up white arrow
334     \@@_assign_delcode:n {"21E9} % down white arrow
335     \@@_assign_delcode:n {"21EA} % up white arrow from bar
336     \@@_assign_delcode:n {"21F3} % updown white arrow
337   }

```

12.4 (Big) operators

The engine does what is necessary to deal with big operators for us automatically with `\Umathchardef`. 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 `_@@_sym:nnn` in the appropriate contexts.

`\l_@@_nolimits_tl` This macro is a sequence containing those maths operators that require a `\nolimits` suffix. This list is used when processing `unicode-math-table.tex` to define

such commands automatically (see the macro `\@@_set_mathsymbol:nNNn`). I've chosen essentially just the operators that look like integrals; hopefully a better mathematician can help me out here. I've a feeling that it's more useful *not* to include the multiple integrals such as \iiint , but that might be a matter of preference.

```

338 \tl_set:Nn \l_@@_nolimits_tl
339   {
340     \int\iint\iiint\iiiint\oint\oiint\oiiint
341     \intclockwise\varointclockwise\ointctrlockwise\sumint
342     \intbar\intBar\fint\cirfnint\awint\rppoint
343     \scpolint\polint\pointint\sqint\intlarhk\intx
344     \intcap\intcup\upint\lowint
345   }

```

12.5 Radicals

`\l_@@_radicals_tl` The radicals are organised in `\@@_set_mathsymbol:nNNn`. We organise radicals in the same way as nolimits-operators. (`\cuberoot` and `\fourthroot`, don't seem to behave as proper radicals.)

```

346 \tl_set:Nn \l_@@_radicals_tl {\sqrt \longdivision \cuberoot \fourthroot}

```

12.6 Fontdimens

```

347 (*LU)

```

`\@@_mathparam_restore:` `\glb@settings` might not be necessary but is included for symmetry. If the maths font were to be loaded later it would clobber our mathparam settings, so this seems like a sensible move.

```

348 \@@_cs_new:Nn \@@_mathparam_restore:
349   {
350     \glb@settings
351     \tl_use:N \g_@@_mathparam_settings_tl
352   }

```

`\@@_mathparam_store:` `\glb@settings` is called to force maths fonts loading *now* so the mathparams are up-to-date.

```

353 \@@_cs_new:Nn \@@_mathparam_store:
354   {
355     \glb@settings
356     \tl_gset:Nx \g_@@_mathparam_settings_tl
357     {
358       \@@_mathparam_store_aux:N \displaystyle
359       \@@_mathparam_store_aux:N \textstyle
360       \@@_mathparam_store_aux:N \scriptstyle
361       \@@_mathparam_store_aux:N \scriptscriptstyle
362     }
363   }
364 \cs_set:Nn \@@_mathparam_store_aux:N
365   {

```

```

366 \Umathquad #1 = \the \Umathquad #1 \scan_stop:
367 \Umathaxis #1 = \the \Umathaxis #1 \scan_stop:
368 \Umathoperatorsize #1 = \the \Umathoperatorsize #1 \scan_stop:
369 \Umathoverbarkern #1 = \the \Umathoverbarkern #1 \scan_stop:
370 \Umathoverbarrule #1 = \the \Umathoverbarrule #1 \scan_stop:
371 \Umathoverbarvgap #1 = \the \Umathoverbarvgap #1 \scan_stop:
372 \Umathunderbarkern #1 = \the \Umathunderbarkern #1 \scan_stop:
373 \Umathunderbarrule #1 = \the \Umathunderbarrule #1 \scan_stop:
374 \Umathunderbarvgap #1 = \the \Umathunderbarvgap #1 \scan_stop:
375 \Umathradicalkern #1 = \the \Umathradicalkern #1 \scan_stop:
376 \Umathradicalrule #1 = \the \Umathradicalrule #1 \scan_stop:
377 \Umathradicalvgap #1 = \the \Umathradicalvgap #1 \scan_stop:
378 \Umathradicaldegreebefore #1 = \the \Umathradicaldegreebefore #1 \scan_stop:
379 \Umathradicaldegreeafter #1 = \the \Umathradicaldegreeafter #1 \scan_stop:
380 \Umathradicaldegreeraise #1 = \the \Umathradicaldegreeraise #1 \scan_stop:
381 \Umathstackvgap #1 = \the \Umathstackvgap #1 \scan_stop:
382 \Umathstacknumup #1 = \the \Umathstacknumup #1 \scan_stop:
383 \Umathstackdenomdown #1 = \the \Umathstackdenomdown #1 \scan_stop:
384 \Umathfractionrule #1 = \the \Umathfractionrule #1 \scan_stop:
385 \Umathfractionnumvgap #1 = \the \Umathfractionnumvgap #1 \scan_stop:
386 \Umathfractionnumup #1 = \the \Umathfractionnumup #1 \scan_stop:
387 \Umathfractiondenomvgap #1 = \the \Umathfractiondenomvgap #1 \scan_stop:
388 \Umathfractiondenomdown #1 = \the \Umathfractiondenomdown #1 \scan_stop:
389 \Umathfractiondelsize #1 = \the \Umathfractiondelsize #1 \scan_stop:
390 \Umathlimitabovevgap #1 = \the \Umathlimitabovevgap #1 \scan_stop:
391 \Umathlimitabovebgap #1 = \the \Umathlimitabovebgap #1 \scan_stop:
392 \Umathlimitabovekern #1 = \the \Umathlimitabovekern #1 \scan_stop:
393 \Umathlimitbelowvgap #1 = \the \Umathlimitbelowvgap #1 \scan_stop:
394 \Umathlimitbelowbgap #1 = \the \Umathlimitbelowbgap #1 \scan_stop:
395 \Umathlimitbelowkern #1 = \the \Umathlimitbelowkern #1 \scan_stop:
396 \Umathoverdelimitervgap #1 = \the \Umathoverdelimitervgap #1 \scan_stop:
397 \Umathoverdelimiterbgap #1 = \the \Umathoverdelimiterbgap #1 \scan_stop:
398 \Umathunderdelimitervgap #1 = \the \Umathunderdelimitervgap #1 \scan_stop:
399 \Umathunderdelimiterbgap #1 = \the \Umathunderdelimiterbgap #1 \scan_stop:
400 \Umathsubshiftdrop #1 = \the \Umathsubshiftdrop #1 \scan_stop:
401 \Umathsubshiftdown #1 = \the \Umathsubshiftdown #1 \scan_stop:
402 \Umathsupshiftdrop #1 = \the \Umathsupshiftdrop #1 \scan_stop:
403 \Umathsupshiftup #1 = \the \Umathsupshiftup #1 \scan_stop:
404 \Umathsubsupshiftdown #1 = \the \Umathsubsupshiftdown #1 \scan_stop:
405 \Umathsubtopmax #1 = \the \Umathsubtopmax #1 \scan_stop:
406 \Umathsupbottommin #1 = \the \Umathsupbottommin #1 \scan_stop:
407 \Umathsupsubbottommax #1 = \the \Umathsupsubbottommax #1 \scan_stop:
408 \Umathsubsupvgap #1 = \the \Umathsubsupvgap #1 \scan_stop:
409 \Umathspaceafterscript #1 = \the \Umathspaceafterscript #1 \scan_stop:
410 \Umathconnectoroverlapmin #1 = \the \Umathconnectoroverlapmin #1 \scan_stop:
411 }

412 </LU>
413 </package>

```

File XII

um-code-fontopt.dtx

13 Font loading options

```
1 (*package)
```

13.1 Math version

```
2 \keys_define:nn {unicode-math}
3 {
4   version .code:n =
5     {
6       \tl_set:Nn \l_@@_mversion_tl {#1}
7       \DeclareMathVersion {\l_@@_mversion_tl}
8     }
9 }
```

13.2 Script and scriptscript font options

```
10 \keys_define:nn {unicode-math}
11 {
12   script-features .tl_set:N = \l_@@_script_features_tl ,
13   sscript-features .tl_set:N = \l_@@_sscript_features_tl ,
14   script-font .tl_set:N = \l_@@_script_font_tl ,
15   sscript-font .tl_set:N = \l_@@_sscript_font_tl ,
16 }
```

13.3 Range processing

Locally redefined all math symbol commands to their slot number prefixed by a quark. Similarly for the math classes.

```
17 \keys_define:nn {unicode-math}
18 {
19   range .code:n =
20     {
21       \bool_if:NF \g_@@_main_font_defined_bool { \@@_error:n {no-main-
font} }
22       \bool_gset_false:N \g_@@_init_bool
23       \@@_range_init:
24       \group_begin:
25       \seq_map_inline:Nn \g_@@_mathclasses_seq
26       {
27         \cs_set:Npn ##1 { \use_none:n \q_unicode_math \exp_not:N ##1 }
28       }
29       \cs_set:Npn \_@@_sym:nnn ##1 ##2 ##3
30       {
31         \cs_set:Npn ##2 { \use_none:n \q_unicode_math ##1 }
32       }

```

```

33     \@@_input_math_symbol_table:
34     \@@_range_process:n {#1}
35     \group_end:
36   }
37 }

```

`\@@_range_init:` Set processing functions if we're not defining the full Unicode math repertoire. Math symbols are defined with `_@@_sym:nnn`; see section §12.1 for the individual definitions

```

38 \@@_cs_new:Nn \@@_range_init:
39 {
40   \int_gincr:N \g_@@_fam_int
41   \tl_set:Nx \l_@@_symfont_label_tl {@@_fam\int_use:N\g_@@_fam_int}
42   \cs_set_eq:NN \_@@_sym:nnn \@@_process_symbol_parse:nnn
43   \cs_set_eq:NN \@@_remap_symbol:nnn \@@_remap_symbol_parse:nnn
44   \cs_set_eq:NN \@@_maybe_init_alphabet:n \use_none:n
45   \cs_set_eq:NN \@@_assign_delcode:nn \@@_assign_delcode_parse:nn
46   \cs_set_eq:NN \@@_make_mathactive:nNN \@@_make_mathactive_parse:nNN

```

Proceed by filling up the various 'range' seqs according to the user options.

```

47   \seq_gclear:N \g_@@_char_range_seq
48   \seq_gclear:N \g_@@_mclass_range_seq
49   \seq_gclear:N \g_@@_mathalph_seq
50 }

```

`\@@_range_process:`

```

51 \cs_new:Nn \@@_range_process:n
52 {
53   \clist_map_inline:nn {#1}
54   {
55     \@@_mathalph_decl:nF {##1} { \@@_range_decl:n {##1} }
56   }
57 }

```

`\@@_mathalph_decl:nF` Possible forms of input:

```

\mathscr
\mathscr->\mathup
\mathscr/{Latin}
\mathscr/{Latin}->\mathup

```

Outputs:

tmpa: math style (e.g., `\mathscr`)

tmpb: alphabets (e.g., Latin)

tmpc: remap style (e.g., `\mathup`). Defaults to tmpa.

The remap style can also be `\mathcal->stixcal`, which I marginally prefer in the general case.

```

58 \cs_new:Nn \@@_mathalph_decl:nF
59 {
60   \tl_set:Nn \l_@@_tmpa_tl {#1}
61   \tl_clear:N \l_@@_tmpb_tl

```

```

62 \tl_clear:N \l_@@_tmpc_tl
63
64 \tl_if_in:NnT \l_@@_tmpa_tl {->}
65 { \exp_after:wN \@@_split_arrow:w \l_@@_tmpa_tl \q_nil }
66
67 \tl_if_in:NnT \l_@@_tmpa_tl {/}
68 { \exp_after:wN \@@_split_slash:w \l_@@_tmpa_tl \q_nil }
69
70 \tl_set:Nx \l_@@_tmpa_tl { \tl_to_str:N \l_@@_tmpa_tl }
71 \exp_args:NNx \tl_remove_all:Nn \l_@@_tmpa_tl { \token_to_str:N \math }
72 \exp_args:NNx \tl_remove_all:Nn \l_@@_tmpa_tl { \token_to_str:N \sym }
73 \tl_trim_spaces:N \l_@@_tmpa_tl
74
75 \tl_if_empty:NT \l_@@_tmpc_tl
76 { \tl_set_eq:NN \l_@@_tmpc_tl \l_@@_tmpa_tl }
77
78 \clist_if_in:NVT \g_@@_bad_alpha_clist \l_@@_tmpa_tl { \@@_er-
ror:n {range-not-bf-sf} }
79
80 \prop_if_exist:cTF {g_@@_named_range_ \l_@@_tmpa_tl _prop}
81 {
82   \seq_gput_right:Nx \g_@@_mathalph_seq
83   {
84     { \exp_not:V \l_@@_tmpa_tl }
85     { \exp_not:V \l_@@_tmpb_tl }
86     { \exp_not:V \l_@@_tmpc_tl }
87   }
88 }
89 {#2}
90 }
91 \cs_set:Npn \@@_split_arrow:w #1->#2 \q_nil
92 {
93   \tl_set:Nx \l_@@_tmpa_tl { \tl_trim_spaces:n {#1} }
94   \tl_set:Nx \l_@@_tmpc_tl { \tl_trim_spaces:n {#2} }
95 }
96 \cs_set:Npn \@@_split_slash:w #1/#2 \q_nil
97 {
98   \tl_set:Nx \l_@@_tmpa_tl { \tl_trim_spaces:n {#1} }
99   \tl_set:Nx \l_@@_tmpb_tl { \tl_trim_spaces:n {#2} }
100 }

```

\@@_range_decl:n

```

101 \cs_new_protected:Nn \@@_range_decl:n
102 {
103   \bool_lazy_and:nnTF { \tl_if_single_p:n {#1} } { \token_if_cs_p:N #1 }
104   % IF A CSNAME:
105   {
106     \tl_if_in:VnTF #1 { \q_unicode_math }
107     {

```



```

108     \seq_if_in:NnTF \g__um_mathclasses_seq {#1}
109     { \seq_gput_right:Nn \g_@@_mclass_range_seq {#1} }
110     { \seq_gput_right:Nx \g_@@_char_range_seq { #1 } }
111   }
112   { \@@_error:nx {bad-cs-in-range} { \tl_to_str:n {#1} } }
113 }
114 % ELSE ASSUME NUMERIC INPUT:
115 {
116   \seq_gput_right:Nx \g_@@_char_range_seq { #1 }
117 }
118 }

```

`\@@_if_char_spec:nNT` #1 : Unicode character slot
 #2 : control sequence (math class)
 #3 : code to execute

This macro expands to #3 if any of its arguments are contained in `\g_@@_char_range_seq`. This list can contain either character ranges (for checking with #1) or control sequences. These latter can either be the command name of a specific character, *or* the math type of one (*e.g.*, `\mathbin`).

Character ranges are passed to `\@@_if_char_spec:nNT`, which accepts input in the form shown in table 1.

Table 1: Ranges accepted by `\@@_if_char_spec:nNT`.

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

We have three tests, performed sequentially in order of execution time. Any test finding a match jumps directly to the end.

```

119 \cs_new:Nn \@@_if_char_spec:nNT
120 {
121   % math class:
122   \seq_if_in:NnT \g_@@_mclass_range_seq {#2}
123   { \use_none_delimit_by_q_nil:w }
124
125   % character slot:
126   \seq_map_inline:Nn \g_@@_char_range_seq
127   {
128     \@@_int_if_slot_is_last_in_range:nnT {#1} {##1}
129     { \seq_gremove_all:Nn \g_@@_char_range_seq {##1} }
130
131     \@@_int_if_slot_in_range:nnT {#1} {##1}
132     { \seq_map_break:n { \use_none_delimit_by_q_nil:w } }
133   }
134

```

```

135 % the following expands to nil if no match was found:
136 \use_none:n
137 \q_nil
138 \use:n
139 {
140   \cs_if_eq:NNT #2 \mathalpha
141   {
142     \clist_put_right:Nx \l_@@_mathmap_charints_clist { \int_eval:n {#1} }
143   }
144   #3
145 }
146 }

```

\@@_int_if_slot_in_range:nnT Pretty basic comma separated range processing. Donald Arseneau's selectp package has a cleverer technique.

A 'numrange' is like -2,5-8,12,17- (can be unsorted).

Four cases, four argument types:

```

% input   #2   #3   #4
% "1 "   [ 1] - [qn] - [ ] qs
% "1- "  [ 1] - [ ] - [qn-] qs
% "-3"   [ ] - [ 3] - [qn-] qs
% "1-3"  [ 1] - [ 3] - [qn-] qs

```

```

147 \cs_new:Nn \@@_int_if_slot_in_range:nnT
148 {
149   \@@_numrange_parse:nwT {#1} #2 - \q_nil - \q_stop {#3}
150 }
151 \cs_set:Npn \@@_numrange_parse:nwT #1 #2 - #3 - #4 \q_stop #5
152 {
153   \tl_if_empty:nTF {#4} { \int_compare:nT {#1=#2} {#5} }
154   {
155     \tl_if_empty:nTF {#3} { \int_compare:nT {#1>=#2} {#5} }
156     {
157       \tl_if_empty:nTF {#2} { \int_compare:nT {#1<=#3} {#5} }
158       {
159         \int_compare:nT {#1>=#2} { \int_compare:nT {#1<=#3} {#5} }
160       } } }
161 }
162 \cs_new:Nn \@@_int_if_slot_is_last_in_range:nnT
163 {
164   \@@_numrange_last_parse:nwT {#1} #2 - \q_nil - \q_stop {#3}
165 }
166 \cs_set:Npn \@@_numrange_last_parse:nwT #1 #2 - #3 - #4 \q_stop #5
167 {
168   \tl_if_empty:nTF {#4} { \int_compare:nT {#1==#2} {#5} }
169   {
170     \tl_if_empty:nTF {#2} { \int_compare:nT {#1==#3} {#5} }
171     {

```

```
172 \int_compare:nT {#1==#3} {#5}
173     } }
174 }
175 </package>
```

File XIII

um-code-fontparam.dtx

14 Cross-platform interface for font parameters

1 (*package)

X_YTeX and LuaTeX have different interfaces for math font parameters. We use LuaTeX's interface because it's much better, but rename the primitives to be more L^AT_EX3-like. There are getter and setter commands for each font parameter. The names of the parameters is derived from the LuaTeX names, with underscores inserted between words. For every parameter $\Umath\langle\text{LuaTeX name}\rangle$, we define an expandable getter command $\@@_\langle\text{L^AT_EX3 name}\rangle:N$ and a protected setter command $\@@_set_\langle\text{L^AT_EX3 name}\rangle:Nn$. The getter command takes one of the style primitives (\displaystyle etc.) and expands to the font parameter, which is a *dimension*. The setter command takes a style primitive and a dimension expression, which is parsed with $\dim_eval:n$.

Often, the mapping between font dimensions and font parameters is bijective, but there are cases which require special attention:

- Some parameters map to different dimensions in display and non-display styles.
- Likewise, one parameter maps to different dimensions in non-cramped and cramped styles.
- There are a few parameters for which X_YTeX doesn't seem to provide \fontdimens ; in this case the getter and setter commands are left undefined.

Cramped style tokens LuaTeX has \crampeddisplaystyle etc., but they are loaded as $\luatexcrampeddisplaystyle$ etc. by the `luatextra` package. X_YTeX, however, doesn't have these primitives, and their syntax cannot really be emulated. Nevertheless, we define these commands as quarks, so they can be used as arguments to the font parameter commands (but nowhere else). Making these commands available is necessary because we need to make a distinction between cramped and non-cramped styles for one font parameter.

$\@@_new_cramped_style:N #1$: command

Define $\langle\text{command}\rangle$ as a new cramped style switch. For LuaTeX, simply rename the corresponding primitive if it is not already defined. For X_YTeX, define $\langle\text{command}\rangle$ as a new quark.

```
2 \cs_new_protected_nopar:Nn \@@_new_cramped_style:N
3 (XE) { \tl_const:Nn #1 { \use_none:n #1 } }
4 (LU) {
5 (LU)   \cs_if_exist:NF #1
6 (LU)     { \cs_new_eq:Nc #1 { luatex \cs_to_str:N #1 } }
7 (LU) }
```

```

\crampeddisplaystyle The cramped style commands.
  \crampedtextstyle 8 \@@_new_cramped_style:N \crampeddisplaystyle
  \crampedscriptstyle 9 \@@_new_cramped_style:N \crampedtextstyle
\crampedscriptscriptstyle 10 \@@_new_cramped_style:N \crampedscriptstyle
  11 \@@_new_cramped_style:N \crampedscriptscriptstyle

```

Font dimension mapping Font parameters may differ between the styles. Lua \TeX accounts for this by having the parameter primitives take a style token argument. To replicate this behavior in \XTeX , we have to map style tokens to specific combinations of font dimension numbers and math fonts (`\textfont` etc.).

```

\@@_font_dimen:Nnnnn #1 : style token
#2 : font dimen for display style
#3 : font dimen for cramped display style
#4 : font dimen for non-display styles
#5 : font dimen for cramped non-display styles
Map math style to  $\XTeX$  math font dimension. <style token> must be one of the style
switches (\displaystyle, \crampeddisplaystyle, ...). The other parameters are
integer constants referring to font dimension numbers. The macro expands to a
dimension which contains the appropriate font dimension.

```

```

12 (*XE)
13 \cs_new_nopar:Npn \@@_font_dimen:Nnnnn #1 #2 #3 #4 #5 {
14   \fontdimen
15   \cs_if_eq:NNTF #1 \displaystyle {
16     #2 \textfont
17   } {
18     \cs_if_eq:NNTF #1 \crampeddisplaystyle {
19       #3 \textfont
20     } {
21       \cs_if_eq:NNTF #1 \textstyle {
22         #4 \textfont
23       } {
24         \cs_if_eq:NNTF #1 \crampedtextstyle {
25           #5 \textfont
26         } {
27           \cs_if_eq:NNTF #1 \scriptstyle {
28             #4 \scriptfont
29           } {
30             \cs_if_eq:NNTF #1 \crampedscriptstyle {
31               #5 \scriptfont
32             } {
33               \cs_if_eq:NNTF #1 \scriptscriptstyle {
34                 #4 \scriptscriptfont
35               } {
36                 #5 \scriptscriptfont
37               }

```

Should we check here if the style is invalid?

```

38         }
39     }
40 }
41 }
42 }
43 }

```

Which family to use?

```

44     2~
45 }
46 </XE>

```

Font parameters This paragraph contains macros for defining the font parameter interface, as well as the definition for all font parameters known to Lua \TeX .

`\@@_font_param:nnnnn` #1 : name

#2 : font dimension for non-cramped display style

#3 : font dimension for cramped display style

#4 : font dimension for non-cramped non-display styles

#5 : font dimension for cramped non-display styles

This macro defines getter and setter functions for the font parameter $\langle name \rangle$. The Lua \TeX font parameter name is produced by removing all underscores and prefixing the result with `Umath`. The X \TeX font dimension numbers must be integer constants.

```

47 \cs_new_protected_nopar:Nn \@@_font_param:nnnnn
48 (*XE)
49 {
50     \@@_font_param_aux:ccnnnn { @@_ #1 :N } { @@_set_ #1 :Nn }
51     { #2 } { #3 } { #4 } { #5 }
52 }
53 </XE>
54 (*LU)
55 {
56     \tl_set:Nn \l_@@_tmpa_tl { #1 }
57     \tl_remove_all:Nn \l_@@_tmpa_tl { _ }
58     \@@_font_param_aux:ccc { @@_ #1 :N } { @@_set_ #1 :Nn }
59     { Umath \l_@@_tmpa_tl }
60 }
61 </LU>

```

`\@@_font_param:nnn` #1 : name

#2 : font dimension for display style

#3 : font dimension for non-display styles

This macro defines getter and setter functions for the font parameter $\langle name \rangle$. The Lua \TeX font parameter name is produced by removing all underscores and prefixing the result with `Umath`. The X \TeX font dimension numbers must be integer constants.

```

62 \cs_new_protected_nopar:Nn \@@_font_param:nnn

```

```

63 {
64   \@@_font_param:nnnn { #1 } { #2 } { #2 } { #3 } { #3 }
65 }

```

\@@_font_param:nn #1 : name

#2 : font dimension

This macro defines getter and setter functions for the font parameter $\langle name \rangle$. The LuaTeX font parameter name is produced by removing all underscores and prefixing the result with U math. The XeTeX font dimension number must be an integer constant.

```

66 \cs_new_protected_nopar:Nn \@@_font_param:nn
67 {
68   \@@_font_param:nnnn { #1 } { #2 } { #2 } { #2 } { #2 }
69 }

```

\@@_font_param:n #1 : name

This macro defines getter and setter functions for the font parameter $\langle name \rangle$, which is considered unavailable in XeTeX. The LuaTeX font parameter name is produced by removing all underscores and prefixing the result with U math.

```

70 \cs_new_protected_nopar:Nn \@@_font_param:n
71 <XE> { }
72 <LU> { \@@_font_param:nnnn { #1 } { 0 } { 0 } { 0 } { 0 } }

```

\@@_font_param_aux:NNnnnn Auxiliary macros for generating font parameter accessor macros.

\@@_font_param_aux:NNN

```

73 <*XE>
74 \cs_new_protected_nopar:Nn \@@_font_param_aux:NNnnnn
75 {
76   \cs_new_nopar:Npn #1 ##1
77   {
78     \@@_font_dimen:Nnnnn ##1 { #3 } { #4 } { #5 } { #6 }
79   }
80   \cs_new_protected_nopar:Npn #2 ##1 ##2
81   {
82     #1 ##1 \dim_eval:n { ##2 }
83   }
84 }
85 \cs_generate_variant:Nn \@@_font_param_aux:NNnnnn { cc }
86 </XE>
87 <*LU>
88 \cs_new_protected_nopar:Nn \@@_font_param_aux:NNN
89 {
90   \cs_new_nopar:Npn #1 ##1
91   {
92     #3 ##1
93   }
94   \cs_new_protected_nopar:Npn #2 ##1 ##2
95   {
96     #3 ##1 \dim_eval:n { ##2 }

```

```

97     }
98   }
99   \cs_generate_variant:Nn \@@_font_param_aux:NNN { ccc }
100  </LU>

```

Now all font parameters that are listed in the LuaTeX reference follow.

```

101 \@@_font_param:nn { axis } { 15 }
102 \@@_font_param:nn { operator_size } { 13 }
103 \@@_font_param:n { fraction_del_size }
104 \@@_font_param:nnn { fraction_denom_down } { 45 } { 44 }
105 \@@_font_param:nnn { fraction_denom_vgap } { 50 } { 49 }
106 \@@_font_param:nnn { fraction_num_up } { 43 } { 42 }
107 \@@_font_param:nnn { fraction_num_vgap } { 47 } { 46 }
108 \@@_font_param:nn { fraction_rule } { 48 }
109 \@@_font_param:nn { limit_above_bgap } { 29 }
110 \@@_font_param:n { limit_above_kern }
111 \@@_font_param:nn { limit_above_vgap } { 28 }
112 \@@_font_param:nn { limit_below_bgap } { 31 }
113 \@@_font_param:n { limit_below_kern }
114 \@@_font_param:nn { limit_below_vgap } { 30 }
115 \@@_font_param:nn { over_delimiter_vgap } { 41 }
116 \@@_font_param:nn { over_delimiter_bgap } { 38 }
117 \@@_font_param:nn { under_delimiter_vgap } { 40 }
118 \@@_font_param:nn { under_delimiter_bgap } { 39 }
119 \@@_font_param:nn { overbar_kern } { 55 }
120 \@@_font_param:nn { overbar_rule } { 54 }
121 \@@_font_param:nn { overbar_vgap } { 53 }
122 \@@_font_param:n { quad }
123 \@@_font_param:nn { radical_kern } { 62 }
124 \@@_font_param:nn { radical_rule } { 61 }
125 \@@_font_param:nnn { radical_vgap } { 60 } { 59 }
126 \@@_font_param:nn { radical_degree_before } { 63 }
127 \@@_font_param:nn { radical_degree_after } { 64 }
128 \@@_font_param:nn { radical_degree_raise } { 65 }
129 \@@_font_param:nn { space_after_script } { 27 }
130 \@@_font_param:nnn { stack_denom_down } { 35 } { 34 }
131 \@@_font_param:nnn { stack_num_up } { 33 } { 32 }
132 \@@_font_param:nnn { stack_vgap } { 37 } { 36 }
133 \@@_font_param:nn { sub_shift_down } { 18 }
134 \@@_font_param:nn { sub_shift_drop } { 20 }
135 \@@_font_param:n { subsup_shift_down }
136 \@@_font_param:nn { sub_top_max } { 19 }
137 \@@_font_param:nn { subsup_vgap } { 25 }
138 \@@_font_param:nn { sup_bottom_min } { 23 }
139 \@@_font_param:nn { sup_shift_drop } { 24 }
140 \@@_font_param:nnnn { sup_shift_up } { 21 } { 22 } { 21 } { 22 }
141 \@@_font_param:nn { supsub_bottom_max } { 26 }
142 \@@_font_param:nn { underbar_kern } { 58 }
143 \@@_font_param:nn { underbar_rule } { 57 }

```



```

144 \@_font_param:nn { underbar_vgap } { 56 }
145 \@_font_param:n { connector_overlap_min }

```

14.1 Historical commands

```

\@@_fontdimen_to_percent:nN #1 : Font dimen number
\@@_fontdimen_pc_to_pt:nN #2 : Font ‘variable’

```

`\fontdimens 10, 11, and 65` aren’t actually dimensions, they’re percentage values given in units of sp. `\@@_fontdimen_to_percent:nn` takes a font dimension number and outputs the decimal value of the associated parameter. `\@@_fontdimen_pc_to_pt:nn` returns a dimension correspond to the current font size relative proportion based on that percentage.

```

146 \cs_set:Nn \@_fontdimen_to_percent:nN
147   {
148     \fp_eval:n { \dim_to_decimal_in_sp:n { \fontdimen #1 #2 } / 100 }
149   }
150 \cs_new:Nn \@_fontdimen_pc_to_pt:nN
151   {
152     \fp_eval:n { \dim_to_decimal_in_sp:n { \fontdimen #1 #2 } / 100 * \f@size }
153   }

```

```

\@@_mathstyle_scale:NnnN #1 : A math style (\scriptstyle, say)
#2 : Macro that takes a non-delimited length argument (like \kern)
#3 : Length control sequence to be scaled according to the math style
#4 : Math font face to use for the lookups

```

This macro is used to scale the lengths reported by `\fontdimen` according to the scale factor for script- and scriptscript-size objects.

```

154 \cs_new:Nn \@_mathstyle_scale:NnnN
155   {
156     \ifx#1\scriptstyle
157       #2 \@_fontdimen_to_percent:nN {10} #4 #3
158     \else
159       \ifx#1\scriptscriptstyle
160         #2 \@_fontdimen_to_percent:nN {11} #4 #3
161       \else
162         #2 #3
163       \fi
164     \fi
165   }
166 </package>

```

File XIV

um-code-mathmap.dtx

15 *Defining the math alphabets per style*

```
1 (*package)
```

`\@@_setup_alphabets:` This function is called within `\setmathfont` to configure the mapping between characters inside math styles. Three modes:

IMPLICIT No ranges specified, set up everything

EXPLICIT Some ranges specified, set up what is requested only

INHERIT Of the slots in the ranges specified, compare against slots in each styled alphabet and only set up those needed

The INHERIT mode saves less time than I was hoping for but is still beneficial in simple cases.

```
2 \@@_cs_new:Nn \@@_setup_alphabets:
3 {
4   \bool_if:NTF \g_@@_init_bool { \@@_setup_alphabets_implicit: }
5   {
6     \seq_if_empty:NF \g_@@_mathalph_seq { \@@_setup_alphabets_explicit: }
7     \clist_if_empty:NF \l_@@_mathmap_charints_clist { \@@_setup_alphabets_inherit: }
8   }
9 }
```

`\@@_setup_alphabets_implicit:`

```
10 \@@_cs_new:Nn \@@_setup_alphabets_implicit:
11 {
12   \@@_log:n {setup-implicit}
13   \seq_gset_eq:NN \g_@@_mathalph_seq \g_@@_default_mathalph_seq
14   \bool_set_true:N \l_@@_implicit_alph_bool
15   \@@_maybe_init_alphabet:n {sf}
16   \@@_maybe_init_alphabet:n {bf}
17   \@@_maybe_init_alphabet:n {bfsf}
18   \cs_set_eq:NN \@@_set_mathalphabet_char:nnn \@@_mathmap_noparse:nnn
19   \cs_set_eq:NN \@@_map_char_single:nn \@@_map_char_noparse:nn
20   \@@_mathalph_map:
21     \seq_if_empty:NF \l_@@_missing_alph_seq { \@@_log:n { missing-
22       alphabets } }
```

`\@@_setup_alphabets_explicit:`

```
23 \@@_cs_new:Nn \@@_setup_alphabets_explicit:
24 {
25   \@@_log:n {setup-explicit}
```

```

26 \bool_set_false:N \l_@@_implicit_alph_bool
27 \cs_set_eq:NN \@@_set_mathalphabet_char:nnn \@@_mathmap_noparse:nnn
28 \cs_set_eq:NN \@@_map_char_single:nn \@@_map_char_noparse:nn
29 \@@_mathalph_map:
30 \seq_if_empty:NF \l_@@_missing_alph_seq { \@@_log:n { missing-
alphabets } }
31 }

```

\@@_setup_alphabets_inherit:

```

32 \@@_cs_new:Nn \@@_setup_alphabets_inherit:
33 {
34 \seq_gclear:N \g_@@_mathalph_seq
35 \seq_map_inline:Nn \g_@@_default_mathalph_seq
36 {
37 \tl_set:No \l_@@_style_tl { \use_i:nnn ##1 }
38 \clist_set:No \l_@@_alphabet_clist { \use_ii:nnn ##1 }
39
40 \clist_map_inline:Nn \l_@@_alphabet_clist
41 {
42 \clist_if_exist:cT {g_@@_named_slots_ \l_@@_style_tl _ #####1 _clist}
43 {
44 \clist_map_inline:cn {g_@@_named_slots_ \l_@@_style_tl _ #####1 _clist}
45 {
46 \clist_map_inline:Nn \l_@@_mathmap_charints_clist
47 {
48 \@@_int_if_slot_in_range:nnT {#####1} {#####1}
49 {
50 \seq_gput_right:Nn \g_@@_mathalph_seq {##1}
51 \clist_map_break:n { \clist_map_break:n { \clist_map_break: } }
52 }
53 }
54 }
55 }
56 }
57 }
58
59 \cs_set_eq:NN \@@_set_mathalphabet_char:nnn \@@_mathmap_parse:nnn
60 \cs_set_eq:NN \@@_map_char_single:nn \@@_map_char_parse:nn
61 \@@_mathalph_map:
62 }

```

\@@_mathalph_map:

```

63 \cs_set:Nn \@@_mathalph_map:
64 {
65 \seq_map_inline:Nn \g_@@_mathalph_seq
66 {
67 \tl_set:No \l_@@_style_tl { \use_i:nnn ##1 }
68 \clist_set:No \l_@@_alphabet_clist { \use_ii:nnn ##1 }
69 \tl_set:No \l_@@_remap_style_tl { \use_iii:nnn ##1 }

```

```

70
71 % If no set of alphabets is defined:
72 \clist_if_empty:NT \l_@@_alphabet_clist
73 {
74   \cs_set_eq:NN \@@_maybe_init_alphabet:n \@@_init_alphabet:n
75   \prop_get:cnN { g_@@_named_range_ \l_@@_style_tl _prop }
76   { default-alpha } \l_@@_alphabet_clist
77 }
78
79 \@@_check_math_alphabet:
80 \@@_setup_math_alphabet:
81 }
82 }

```

`\@@_check_math_alphabet:` First check that at least one of the alphabets for the font shape is defined (this process is fast) ...

```

83 \cs_new:Nn \@@_check_math_alphabet:
84 {
85   \clist_map_inline:Nn \l_@@_alphabet_clist
86   {
87     \tl_set:Nn \l_@@_alphabet_tl {##1}
88     \@@_if_alphabet_exists:nnTF \l_@@_style_tl \l_@@_alphabet_tl
89     {
90       \str_if_eq:eeTF {\l_@@_alphabet_tl} {misc}
91       {
92         \@@_maybe_init_alphabet:n \l_@@_style_tl
93         \clist_map_break:
94       }
95       {
96         \@@_glyph_if_exist:NnT \g_@@_curr_font_cmd_tl
97         { \@@_to_usv:nn {\l_@@_style_tl} {\l_@@_alphabet_tl} }
98         {
99           \@@_maybe_init_alphabet:n \l_@@_style_tl
100          \clist_map_break:
101        }
102      }
103    }
104    {
105      \msg_warning:nxx {unicode-math} {no-alphabet}
106      { \l_@@_style_tl / \l_@@_alphabet_tl }
107    }
108  }
109 }

```

`\@@_setup_math_alphabet:` ...and then loop through them defining the individual ranges: (currently this process is slow)

```

110 \@@_cs_new:Nn \@@_setup_math_alphabet:
111 {
112   \clist_map_inline:Nn \l_@@_alphabet_clist

```

```

113     {
114         \tl_set:Nx \l_@@_alphabet_tl { \tl_trim_spaces:n {##1} }
115
116     (debug)\@@_debug:n {_setup_math_alphabet:~\l_@@_style_tl/\l_@@_alphabet_tl}
117
118     \@@_if_alphabet_exists:nnT {\l_@@_style_tl} {\l_@@_alphabet_tl}
119     {
120         \exp_args:No \tl_if_eq:nnTF \l_@@_alphabet_tl {misc}
121         {
122             \@@_log:nx {setup- $\alpha$ } {sym \l_@@_style_tl~(\l_@@_alphabet_tl)}
123             \@@_alphabet_config:nnn {\l_@@_style_tl} {\l_@@_alphabet_tl} {\l_@@_remap_style_tl}
124         }
125         {
126             \@@_glyph_if_exist:NnTF \g_@@_curr_font_cmd_tl { \@@_to_usv:nn {\l_@@_remap_style_tl} }
127             {
128                 \@@_log:nx {setup- $\alpha$ } {sym \l_@@_style_tl~(\l_@@_alphabet_tl)}
129                 \@@_alphabet_config:nnn {\l_@@_style_tl} {\l_@@_alphabet_tl} {\l_@@_remap_style_tl}
130             }
131             {
132                 \bool_if:NTF \l_@@_implicit_alph_bool
133                 {
134                     \seq_put_right:Nx \l_@@_missing_alph_seq
135                     {
136                         \backslashchar sym \l_@@_style_tl \space
137                         (\tl_use:c{c_@@_math_alphabet_name_ \l_@@_alphabet_tl _tl})
138                     }
139                 }
140                 {
141                     \@@_alphabet_config:nnn {\l_@@_style_tl} {\l_@@_alphabet_tl} {up}
142                 }
143             }
144         }
145     }
146 }
147 }

```

Each alphabet style needs to be configured. This happens in Section 17.

```

148 \cs_new:Nn \@@_new_alphabet_config:nnn
149 {
150     \prop_if_exist:cF {g_@@_named_range_#1_prop}
151     { \@@_warning:nnn {no-named-range} {#1} {#2} }
152
153
154     \prop_gput:cnn {g_@@_named_range_#1_prop} { alpha_tl }
155     {
156         \prop_item:cn {g_@@_named_range_#1_prop} { alpha_tl } {#2}
157     }
158     % Q: do I need to bother removing duplicates?
159

```

Create list of all chars defined in this named range:

```

160 \cs_new:cn { @@_config_#1_#2:n }
161 {
162   \clist_gclear_new:c {g_@@_named_slots_#1_#2_clist}
163   \tl_set:Nn \l_@@_curr_named_slot { g_@@_named_slots_#1_#2_clist }
164   #3
165   \clist_gremove_duplicates:c {g_@@_named_slots_#1_#2_clist}
166 }
167
168 }
169 \cs_new:Nn \@@_alphabet_config:nnn
170 {
171   \use:c {@@_config_#1_#2:n} {#3}
172 }
173 \prg_new_conditional:Nnn \@@_if_alphabet_exists:nn {T,TF}
174 {
175   \cs_if_exist:cTF {@@_config_#1_#2:n}
176   \prg_return_true: \prg_return_false:
177 }

```

15.1 Mapping ‘naked’ math characters

Before we show the definitions of the alphabet mappings using the functions `\@@_alphabet_config:nnn \l_@@_style_tl {##1} {...}`, we first want to define some functions to be used inside them to actually perform the character mapping.

15.1.1 Functions

`\@@_map_char_single:nn` Wrapper for `\@@_map_char_noparse:nn` or `\@@_map_char_parse:nn` depending on the context.

```

\@@_map_char_noparse:nn
\@@_map_char_parse:nn 178 \cs_new:Nn \@@_map_char_noparse:nn
179 {
180   \@@_set_mathcode:nnnn {#1} {\mathalpha} {\l_@@_symfont_label_tl} {#2}
181 }
182 \cs_new:Nn \@@_map_char_parse:nn
183 {
184   \@@_if_char_spec:nNT {#1} {\mathalpha}
185   { \@@_map_char_noparse:nn {#1}{#2} }
186 }

```

`\@@_map_char_single:nnn` #1 : char name (‘dotlessi’)
 #2 : from alphabet(s)
 #3 : to alphabet
 Logical interface to `\@@_map_char_single:nn`.

```

187 \cs_new:Nn \@@_map_char_single:nnn

```

```

188 {
189   \@@_map_char_single:nn { \@@_to_usv:nn {#1} {#3} }
190                           { \@@_to_usv:nn {#2} {#3} }
191 }

```

\@@_map_chars_range:nnnn #1 : Number of chars (26)
 #2 : From style, one or more (it)
 #3 : To style (up)
 #4 : Alphabet name (Latin)

First the function with numbers:

```

192 \cs_set:Nn \@@_map_chars_range:nnn
193 {
194   \int_step_inline:nnnn {0} {1} {#1-1}
195     { \@@_map_char_single:nn {#2+##1} {#3+##1} }
196
197   \clist_gput_right:cx { \l_@@_curr_named_slot }
198     { \int_eval:n { #3 } - \int_eval:n { #3 + #1-1 } }
199 }

```

And the wrapper with names:

```

200 \cs_new:Nn \@@_map_chars_range:nnnn
201 {
202   \@@_map_chars_range:nnn {#1} { \@@_to_usv:nn {#2} {#4} }
203                           { \@@_to_usv:nn {#3} {#4} }
204 }

```

15.1.2 Functions for ‘normal’ alphabet symbols

\@@_set_normal_char:nnn

```

205 \cs_set:Nn \@@_set_normal_char:nnn
206 {
207   \@@_usv_if_exist:nnT {#3} {#1}
208   {
209     \clist_map_inline:nn {#2}
210     {
211       \@@_set_mathalphabet_pos:nnnn {normal} {#1} {##1} {#3}
212       \@@_map_char_single:nnn {##1} {#3} {#1}
213
214       \clist_gput_right:cx { \l_@@_curr_named_slot }
215         { \int_eval:n { \@@_to_usv:nn {#3} {#1} } }
216     }
217   }
218 }

```

```

219 \cs_new:Nn \@@_set_normal_Latin:nn
220 {
221   \clist_map_inline:nn {#1}
222   {
223     \@@_set_mathalphabet_Latin:nnn {normal} {##1} {#2}

```

```

224     \@@_map_chars_range:nnnn {26} {##1} {#2} {Latin}
225   }
226 }
227 \cs_new:Nn \@@_set_normal_latin:nn
228 {
229   \clist_map_inline:nn {#1}
230   {
231     \@@_set_mathalphabet_latin:nnn {normal} {##1} {#2}
232     \@@_map_chars_range:nnnn {26} {##1} {#2} {latin}
233   }
234 }
235 \cs_new:Nn \@@_set_normal_greek:nn
236 {
237   \clist_map_inline:nn {#1}
238   {
239     \@@_set_mathalphabet_greek:nnn {normal} {##1} {#2}
240     \@@_map_chars_range:nnnn {25} {##1} {#2} {greek}
241     \@@_map_char_single:nnn {##1} {#2} {epsilon}
242     \@@_map_char_single:nnn {##1} {#2} {vartheta}
243     \@@_map_char_single:nnn {##1} {#2} {varkappa}
244     \@@_map_char_single:nnn {##1} {#2} {phi}
245     \@@_map_char_single:nnn {##1} {#2} {varrho}
246     \@@_map_char_single:nnn {##1} {#2} {varpi}
247     \@@_set_mathalphabet_pos:nnnn {normal} {epsilon} {##1} {#2}
248     \@@_set_mathalphabet_pos:nnnn {normal} {vartheta} {##1} {#2}
249     \@@_set_mathalphabet_pos:nnnn {normal} {varkappa} {##1} {#2}
250     \@@_set_mathalphabet_pos:nnnn {normal} {phi} {##1} {#2}
251     \@@_set_mathalphabet_pos:nnnn {normal} {varrho} {##1} {#2}
252     \@@_set_mathalphabet_pos:nnnn {normal} {varpi} {##1} {#2}
253   }
254 }
255 \cs_new:Nn \@@_set_normal_Greek:nn
256 {
257   \clist_map_inline:nn {#1}
258   {
259     \@@_set_mathalphabet_Greek:nnn {normal} {##1} {#2}
260     \@@_map_chars_range:nnnn {25} {##1} {#2} {Greek}
261     \@@_map_char_single:nnn {##1} {#2} {varTheta}
262     \@@_set_mathalphabet_pos:nnnn {normal} {varTheta} {##1} {#2}
263   }
264 }
265 \cs_new:Nn \@@_set_normal_numbers:nn
266 {
267   \@@_set_mathalphabet_numbers:nnn {normal} {#1} {#2}
268   \@@_map_chars_range:nnnn {10} {##1} {#2} {num}
269 }

```


15.2 Mapping chars inside a math style

15.2.1 Functions for setting up the maths alphabets

```
\@@_set_mathalphabet_char:nnn #1 : Maths alphabet, e.g., 'bb'  
#2 : Input slot, e.g., the slot for 'A' (comma separated)  
#3 : Output slot, e.g., the slot for 'A'  
This is a wrapper for either \@@_mathmap_noparse:nnn or \@@_mathmap_parse:nnn,  
depending on the context.  
  
\@@_mathmap_noparse:nnn #1 : Maths alphabet, e.g., 'bb'  
#2 : Input slot, e.g., the slot for 'A' (comma separated)  
#3 : Output slot, e.g., the slot for 'A'  
Adds \@@_set_mathcode:nnnn declarations to the specified maths alphabet's def-  
inition.  
270 \cs_new:Nn \@@_mathmap_noparse:nnn  
271 {  
272   \tl_gput_right:cx { g_@@_switchto_#1_tl }  
273   {  
274     \@@_set_mathcode:nnnn {#2} {\mathalpha} {\l_@@_symfont_label_tl} {#3}  
275   }  
276 }  
  
\@@_mathmap_parse:nnn #1 : Maths alphabet, e.g., 'bb'  
#2 : Input slot, e.g., the slot for 'A' (comma separated)  
#3 : Output slot, e.g., the slot for 'A'  
When \@@_if_char_spec:nNT is executed, it populates the \l_@@_mathmap_-  
charints_clist macro with slot numbers corresponding to the specified range. This  
range is used to conditionally add \@@_set_mathcode:nnnn declarations to the  
maths alphabet definition.  
277 \cs_new:Nn \@@_mathmap_parse:nnn  
278 {  
279   \exp_args:NNx \clist_if_in:NnT \l_@@_mathmap_charints_clist { \int_eval:n {#3} }  
280   {  
281     \@@_mathmap_noparse:nnn {#1} {#2} {#3}  
282   }  
283 }  
  
\@@_set_mathalphabet_char:nnnn #1 : math style command  
#2 : input math alphabet name  
#3 : output math alphabet name  
#4 : char name to map  
284 \cs_new:Nn \@@_set_mathalphabet_char:nnnn  
285 {  
286   \@@_set_mathalphabet_char:nnn {#1} { \@@_to_usv:nn {#2} {#4} }  
287   { \@@_to_usv:nn {#3} {#4} }  
288 }
```

`\@@_set_mathalph_range:nnnn` #1 : Number of iterations
 #2 : Sym command suffix
 #3 : Starting input char
 #4 : Starting output char
 Loops through character ranges setting `\mathcode`. First the version that uses numbers:

```
289 \cs_new:Nn \@@_set_mathalph_range:nnnn
290 {
291   \int_step_inline:nnnn {0} {1} {#1-1}
292     { \@@_set_mathalphabet_char:nnn {#2} { ##1 + #3 } { ##1 + #4 } }
293 }
```

`\@@_set_mathalph_range:nnnn` #1 : Number of iterations
 #2 : Sym command suffix
 #3 : input style
 #4 : output style
 #5 : alphabet
 Then the wrapper version that uses names:

```
294 \cs_new:Nn \@@_set_mathalph_range:nnnnn
295 {
296   \clist_gput_right:cx { \l_@@_curr_named_slot }
297     { \int_eval:n { \@@_to_usv:nn {#4} {#5} } - \int_eval:n { (#1-
298     1)+\@@_to_usv:nn {#4} {#5} } }
299   \@@_set_mathalph_range:nnnn {#1} {#2} { \@@_to_usv:nn {#3} {#5} }
300     { \@@_to_usv:nn {#4} {#5} }
301 }
```

15.2.2 Individual mapping functions for different alphabets

```
302 \cs_new:Nn \@@_set_mathalphabet_pos:nnnn
303 {
304   \@@_usv_if_exist:nnT {#4} {#2}
305     {
306       \clist_map_inline:nn {#3}
307         { \@@_set_mathalphabet_char:nnnn {#1} {##1} {#4} {#2} }
308     }
309   \clist_gput_right:cx { \l_@@_curr_named_slot }
310     { \int_eval:n { \@@_to_usv:nn {#4} {#2} } }
311 }
312 }
313 \cs_new:Nn \@@_set_mathalphabet_numbers:nnn
314 {
315   \clist_map_inline:nn {#2}
316     { \@@_set_mathalph_range:nnnnn {10} {#1} {##1} {#3} {num} }
317 }
318 \cs_new:Nn \@@_set_mathalphabet_Latin:nnn
319 {
```

```

320 \clist_map_inline:nn {#2}
321   { \@@_set_mathalph_range:nnnn {26} {#1} {##1} {#3} {Latin} }
322 }
323 \cs_new:Nn \@@_set_mathalphabet_latin:nnn
324   {
325   \clist_map_inline:nn {#2}
326     {
327     \@@_set_mathalph_range:nnnn {26} {#1} {##1} {#3} {latin}
328     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {h}
329     }
330   }
331 \cs_new:Nn \@@_set_mathalphabet_Greek:nnn
332   {
333   \clist_map_inline:nn {#2}
334     {
335     \@@_set_mathalph_range:nnnn {25} {#1} {##1} {#3} {Greek}
336     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {varTheta}
337     }
338   }
339 \cs_new:Nn \@@_set_mathalphabet_greek:nnn
340   {
341   \clist_map_inline:nn {#2}
342     {
343     \@@_set_mathalph_range:nnnn {25} {#1} {##1} {#3} {greek}
344     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {epsilon}
345     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {vartheta}
346     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {varkappa}
347     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {phi}
348     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {varrho}
349     \@@_set_mathalphabet_char:nnnn {#1} {##1} {#3} {varpi}
350     }
351   }
352 </package>

```

File XV

um-code-sym-commands.dtx

16 Mapping in maths alphabets

1 (*package)

16.1 Setting styles

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

Implicit mode:

- Try and set all of the alphabet shapes.
- Check for the first glyph of each alphabet to detect if the font supports each alphabet shape.
- For alphabets that do exist, overwrite whatever's already there.
- For alphabets that are not supported, *do nothing*. (This includes leaving the old alphabet definition in place.)

Explicit mode:

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

16.2 Defining the math style macros

We call the different shapes that a math alphabet can be a 'math style'. Note that different alphabets can exist within the same math style. E.g., we call 'bold' the math style `bf` and within it there are upper and lower case Greek and Roman alphabets and Arabic numerals.

`\@@_prepare_mathstyle:n` #1 : math style name (e.g., `it` or `bb`)

Define the high level math alphabet macros (`\mathit`, etc.) in terms of unicode-math definitions. Use `\bgroup/\egroup` so `s'scripts` scan the whole thing.

The flag `\l_@@_mathstyle_tl` is for other applications to query the current math style.

```
2 \@@_cs_new:Nn \@@_prepare_mathstyle:n
3 {
```

```

4 \seq_gput_right:Nn \g_@@_mathstyles_seq {#1}
5 \@@_init_alphabet:n {#1}
6 \cs_set_protected:cpx {sym#1}
7 {
8   \@@_group_begin:
9   \exp_not:n
10  {
11    \mode_if_math:F { \exp_args:Nc \non@alpherr {sym#1} }
12    \tl_set:Nn \l_@@_mathstyle_tl {#1}
13  }
14  \@@_switch_to:n {#1}
15  \@@_mathgroup_set:n {-1}
16  \@@_group_end:n
17 }
18 }

```

`\@@_init_alphabet:n #1` : math alphabet name (e.g., `it` or `bb`)

This macro initialises the macros used to set up a math alphabet. First used when the math alphabet macro is first defined, but then used later when redefining a particular maths alphabet.

```

19 \@@_cs_new:Nn \@@_init_alphabet:n
20 {
21   \@@_log:nx {alph-initialise} {#1}
22   \tl_gclear_new:c {g_@@_switchto_#1_tl}
23   \cs_set_protected:cpn {@@_switchto_#1:} { \tl_use:c {g_@@_switchto_#1_tl} }
24 }
25 \cs_new_protected:Nn \@@_switch_to:n
26 {
27   \tl_use:c {g_@@_switchto_#1_tl}
28 }

```

16.3 Definition of alphabets and styles

The linking between named ranges and symbol style commands happens here. It's currently not using all of the machinery we're in the process of setting up above. Baby steps.

```

29 \@@_cs_new:Nn \@@_default_mathalph:nnn
30 {
31   \prop_new:c {g_@@_named_range_#1_prop}
32   \seq_gput_right:Nn \g_@@_default_mathalph_seq {{#1}{#2}{#3}}
33   \prop_gput:cnn { g_@@_named_range_#1_prop } { default-alpha } {#2}
34 }
35 \@@_default_mathalph:nnn {up } {latin, Latin, greek, Greek, num, misc} {up }
36 \@@_default_mathalph:nnn {it } {latin, Latin, greek, Greek, misc} {it }
37 \@@_default_mathalph:nnn {bb } {latin, Latin, num, misc} {bb }
38 \@@_default_mathalph:nnn {bbit } {misc} {bbit }
39 \@@_default_mathalph:nnn {scr } {latin, Latin} {scr }
40 \@@_default_mathalph:nnn {cal } {latin, Latin} {scr }

```

```

41 \@@_default_mathalph:nnn {bfc al } {latin, Latin}           {bfscr }
42 \@@_default_mathalph:nnn {frak } {latin, Latin}           {frak }
43 \@@_default_mathalph:nnn {tt } {latin, Latin, num}        {tt }
44 \@@_default_mathalph:nnn {sfup } {latin, Latin, num}      {sfup }
45 \@@_default_mathalph:nnn {sfit } {latin, Latin}           {sfit }
46 \@@_default_mathalph:nnn {bfup } {latin, Latin, greek, Greek, num, misc} {bfup }
47 \@@_default_mathalph:nnn {bfit } {latin, Latin, greek, Greek, misc} {bfit }
48 \@@_default_mathalph:nnn {bfscr } {latin, Latin}          {bfscr }
49 \@@_default_mathalph:nnn {bffrak} {latin, Latin}          {bffrak}
50 \@@_default_mathalph:nnn {bfsfup} {latin, Latin, greek, Greek, num, misc} {bfs-
fup}
51 \@@_default_mathalph:nnn {bfsfit} {latin, Latin, greek, Greek, misc} {bfs-
fit}

```

16.3.1 Define symbol style commands

Finally, all of the ‘symbol styles’ commands are set up, which are the commands to access each of the named alphabet styles. There is not a one-to-one mapping between symbol style commands and named style ranges!

```

52 \clist_map_inline:nn
53 {
54   up, it, bfup, bfit, sfup, sfit, bfsfup, bfsfit, bfsf,
55   tt, bb, bbit, scr, bfscr, cal, bfc al, frak, bffrak,
56   normal, literal, sf, bf,
57 }
58 {
59   \@@_prepare_mathstyle:n {#1}
60 }

```

16.3.2 New names for legacy textmath alphabet selection

In case a package option overwrites, say, `\mathbf` with `\ymbf`.

```

61 \clist_map_inline:nn
62 { rm, it, bf, sf, tt }
63 { \cs_set_eq:cc { mathtext #1 } { math #1 } }

```

Perhaps these should actually be defined using a hypothetical unicode-math interface to creating new such styles. To come.

16.3.3 Replacing legacy pure-maths alphabets

The following are alphabets which do not have a math/text ambiguity.

```

64 \clist_map_inline:nn
65 {
66   normal, bb, bbit, scr, bfscr, cal, bfc al, frak, bffrak, tt,
67   bfup, bfit, sfup, sfit, bfsfup, bfsfit, bfsf
68 }
69 {
70   \cs_set:cpx { math #1 } { \exp_not:c { sym #1 } }
71 }

```

16.3.4 *New commands for ambiguous alphabets*

```
72 \AtBeginDocument { \@@_setup_mathtext: }
73 \@@_cs_new:Nn \@@_setup_mathtext:
74 {
75   \clist_map_inline:nn
76     { rm, it, bf, sf, tt }
77     {
78       \cs_set_protected:cpx { math ##1 }
79       {
80         \exp_not:n { \bool_if:NTF } \exp_not:c { g_@@_ math ##1 _text_bool }
81         { \exp_not:c { mathtext ##1 } }
82         { \exp_not:c { sym ##1 } }
83       }
84     }
85 }
```

Alias `\mathrm` as legacy name for `\mathup`

```
86 \cs_set_protected:Npn \mathup { \mathrm }
87 \cs_set_protected:Npn \symrm { \symup }
88 </package>
```

File XVI

um-code-alphabets.dtx

17 *Setting up alphabets*

1 *(*package)*

17.1 *Upright: up*

```
2 \@@_new_alphabet_config:nnn {up} {num}
3 {
4   \@@_set_normal_numbers:nn {up} {#1}
5   \@@_set_mathalphabet_numbers:nnn {up} {up} {#1}
6 }
7
8 \@@_new_alphabet_config:nnn {up} {Latin}
9 {
10  \bool_if:NTF \g_@@_literal_bool { \@@_set_normal_Latin:nn {up} {#1} }
11  {
12    \bool_if:NT \g_@@_upLatin_bool { \@@_set_normal_Latin:nn {up,it} {#1} }
13  }
14  \@@_set_mathalphabet_Latin:nnn {up} {up,it} {#1}
15  \@@_set_mathalphabet_Latin:nnn {literal} {up} {up}
16  \@@_set_mathalphabet_Latin:nnn {literal} {it} {it}
17 }
18
19 \@@_new_alphabet_config:nnn {up} {latin}
20 {
21  \bool_if:NTF \g_@@_literal_bool { \@@_set_normal_latin:nn {up} {#1} }
22  {
23    \bool_if:NT \g_@@_uplatin_bool
24    {
25      \@@_set_normal_latin:nn {up,it} {#1}
26      \@@_set_normal_char:nnn {h} {up,it} {#1}
27      \@@_set_normal_char:nnn {dotlessi} {up,it} {#1}
28      \@@_set_normal_char:nnn {dotlessj} {up,it} {#1}
29    }
30  }
31  \@@_set_mathalphabet_latin:nnn {up} {up,it}{#1}
32  \@@_set_mathalphabet_latin:nnn {literal} {up} {up}
33  \@@_set_mathalphabet_latin:nnn {literal} {it} {it}
34 }
35
36 \@@_new_alphabet_config:nnn {up} {Greek}
37 {
38  \bool_if:NTF \g_@@_literal_bool { \@@_set_normal_Greek:nn {up}{#1} }
39  {
40    \bool_if:NT \g_@@_upGreek_bool { \@@_set_normal_Greek:nn {up,it}{#1} }
```



```

41     }
42     \@@_set_mathalphabet_Greek:nnn {up} {up,it}{#1}
43     \@@_set_mathalphabet_Greek:nnn {literal} {up} {up}
44     \@@_set_mathalphabet_Greek:nnn {literal} {it} {it}
45 }
46
47 \@@_new_alphabet_config:nnn {up} {greek}
48 {
49     \bool_if:NTF \g_@@_literal_bool { \@@_set_normal_greek:nn {up} {#1} }
50     {
51         \bool_if:NT \g_@@_upgreek_bool
52         {
53             \@@_set_normal_greek:nn {up,it} {#1}
54         }
55     }
56     \@@_set_mathalphabet_greek:nnn {up} {up,it} {#1}
57     \@@_set_mathalphabet_greek:nnn {literal} {up} {up}
58     \@@_set_mathalphabet_greek:nnn {literal} {it} {it}
59 }
60
61 \@@_new_alphabet_config:nnn {up} {misc}
62 {
63     \bool_if:NTF \g_@@_literal_Nabla_bool
64     {
65         \@@_set_normal_char:nnn {Nabla}{up}{up}
66     }
67     {
68         \bool_if:NT \g_@@_upNabla_bool
69         {
70             \@@_set_normal_char:nnn {Nabla}{up,it}{up}
71         }
72     }
73     \bool_if:NTF \g_@@_literal_partial_bool
74     {
75         \@@_set_normal_char:nnn {partial}{up}{up}
76     }
77     {
78         \bool_if:NT \g_@@_uppartial_bool
79         {
80             \@@_set_normal_char:nnn {partial}{up,it}{up}
81         }
82     }
83     \@@_set_mathalphabet_pos:nnnn {up} {partial} {up,it} {#1}
84     \@@_set_mathalphabet_pos:nnnn {up} {Nabla} {up,it} {#1}
85     \@@_set_mathalphabet_pos:nnnn {up} {dotlessi} {up,it} {#1}
86     \@@_set_mathalphabet_pos:nnnn {up} {dotlessj} {up,it} {#1}
87 }

```

17.2 *Italic: it*

```

88 \@@_new_alphabet_config:nnn {it} {Latin}
89 {
90   \bool_if:NTF \g_@@_literal_bool { \@@_set_normal_Latin:nn {it} {#1} }
91   {
92     \bool_if:NF \g_@@_upLatin_bool { \@@_set_normal_Latin:nn {up,it} {#1} }
93   }
94   \@@_set_mathalphabet_Latin:nnn {it} {up,it} {#1}
95 }
96
97 \@@_new_alphabet_config:nnn {it} {latin}
98 {
99   \bool_if:NTF \g_@@_literal_bool
100   {
101     \@@_set_normal_latin:nn {it}{#1}
102     \@@_set_normal_char:nnn {h}{it}{#1}
103   }
104   {
105     \bool_if:NF \g_@@_uplatin_bool
106     {
107       \@@_set_normal_latin:nn {up,it} {#1}
108       \@@_set_normal_char:nnn {h} {up,it} {#1}
109       \@@_set_normal_char:nnn {dotlessi} {up,it} {#1}
110       \@@_set_normal_char:nnn {dotlessj} {up,it} {#1}
111     }
112   }
113   \@@_set_mathalphabet_latin:nnn {it} {up,it} {#1}
114   \@@_set_mathalphabet_pos:nnnn {it} {dotlessi} {up,it} {#1}
115   \@@_set_mathalphabet_pos:nnnn {it} {dotlessj} {up,it} {#1}
116 }
117
118 \@@_new_alphabet_config:nnn {it} {Greek}
119 {
120   \bool_if:NTF \g_@@_literal_bool
121   {
122     \@@_set_normal_Greek:nn {it} {#1}
123   }
124   {
125     \bool_if:NF \g_@@_upGreek_bool { \@@_set_normal_Greek:nn {up,it} {#1} }
126   }
127   \@@_set_mathalphabet_Greek:nnn {it} {up,it} {#1}
128 }
129
130 \@@_new_alphabet_config:nnn {it} {greek}
131 {
132   \bool_if:NTF \g_@@_literal_bool
133   {
134     \@@_set_normal_greek:nn {it} {#1}
135   }
136   {

```

```

137     \bool_if:NF \g_@@_upgreek_bool { \@@_set_normal_greek:nn {it,up} {#1} }
138   }
139   \@@_set_mathalphabet_greek:nnn {it} {up,it} {#1}
140 }
141
142 \@@_new_alphabet_config:nnn {it} {misc}
143 {
144   \bool_if:NTF \g_@@_literal_Nabla_bool
145   {
146     \@@_set_normal_char:nnn {Nabla} {it} {it}
147   }
148   {
149     \bool_if:NF \g_@@_upNabla_bool
150     {
151       \@@_set_normal_char:nnn {Nabla} {up,it} {it}
152     }
153   }
154   \bool_if:NTF \g_@@_literal_partial_bool
155   {
156     \@@_set_normal_char:nnn {partial} {it} {it}
157   }
158   {
159     \bool_if:NF \g_@@_uppartial_bool
160     {
161       \@@_set_normal_char:nnn {partial} {up,it} {it}
162     }
163   }
164   \@@_set_mathalphabet_pos:nnnn {it} {partial} {up,it}{#1}
165   \@@_set_mathalphabet_pos:nnnn {it} {Nabla} {up,it}{#1}
166 }

```

17.3 Blackboard or double-struck: *bb* and *bbit*

```

167 \@@_new_alphabet_config:nnn {bb} {latin}
168 {
169   \@@_set_mathalphabet_latin:nnn {bb} {up,it} {#1}
170 }
171
172 \@@_new_alphabet_config:nnn {bb} {Latin}
173 {
174   \@@_set_mathalphabet_Latin:nnn {bb} {up,it} {#1}
175   \@@_set_mathalphabet_pos:nnnn {bb} {C} {up,it} {#1}
176   \@@_set_mathalphabet_pos:nnnn {bb} {H} {up,it} {#1}
177   \@@_set_mathalphabet_pos:nnnn {bb} {N} {up,it} {#1}
178   \@@_set_mathalphabet_pos:nnnn {bb} {P} {up,it} {#1}
179   \@@_set_mathalphabet_pos:nnnn {bb} {Q} {up,it} {#1}
180   \@@_set_mathalphabet_pos:nnnn {bb} {R} {up,it} {#1}
181   \@@_set_mathalphabet_pos:nnnn {bb} {Z} {up,it} {#1}
182 }
183

```

```

184 \@@_new_alphabet_config:nnn {bb} {num}
185 {
186   \@@_set_mathalphabet_numbers:nnn {bb} {up} {#1}
187 }
188
189 \@@_new_alphabet_config:nnn {bb} {misc}
190 {
191   \@@_set_mathalphabet_pos:nnnn {bb}      {Pi} {up,it} {#1}
192   \@@_set_mathalphabet_pos:nnnn {bb}      {pi} {up,it} {#1}
193   \@@_set_mathalphabet_pos:nnnn {bb}      {Gamma} {up,it} {#1}
194   \@@_set_mathalphabet_pos:nnnn {bb}      {gamma} {up,it} {#1}
195   \@@_set_mathalphabet_pos:nnnn {bb} {summation} {up} {#1}
196 }
197
198 \@@_new_alphabet_config:nnn {bbit} {misc}
199 {
200   \@@_set_mathalphabet_pos:nnnn {bbit} {D} {up,it} {#1}
201   \@@_set_mathalphabet_pos:nnnn {bbit} {d} {up,it} {#1}
202   \@@_set_mathalphabet_pos:nnnn {bbit} {e} {up,it} {#1}
203   \@@_set_mathalphabet_pos:nnnn {bbit} {i} {up,it} {#1}
204   \@@_set_mathalphabet_pos:nnnn {bbit} {j} {up,it} {#1}
205 }

```

17.4 *Script and calligraphic: scr and cal*

```

206 \@@_new_alphabet_config:nnn {scr} {Latin}
207 {
208   \@@_set_mathalphabet_Latin:nnn {scr}      {up,it} {#1}
209   \@@_set_mathalphabet_pos:nnnn {scr} {B} {up,it} {#1}
210   \@@_set_mathalphabet_pos:nnnn {scr} {E} {up,it} {#1}
211   \@@_set_mathalphabet_pos:nnnn {scr} {F} {up,it} {#1}
212   \@@_set_mathalphabet_pos:nnnn {scr} {H} {up,it} {#1}
213   \@@_set_mathalphabet_pos:nnnn {scr} {I} {up,it} {#1}
214   \@@_set_mathalphabet_pos:nnnn {scr} {L} {up,it} {#1}
215   \@@_set_mathalphabet_pos:nnnn {scr} {M} {up,it} {#1}
216   \@@_set_mathalphabet_pos:nnnn {scr} {R} {up,it} {#1}
217 }
218
219 \@@_new_alphabet_config:nnn {scr} {latin}
220 {
221   \@@_set_mathalphabet_latin:nnn {scr}      {up,it} {#1}
222   \@@_set_mathalphabet_pos:nnnn {scr} {e} {up,it} {#1}
223   \@@_set_mathalphabet_pos:nnnn {scr} {g} {up,it} {#1}
224   \@@_set_mathalphabet_pos:nnnn {scr} {o} {up,it} {#1}
225 }

```

These are by default synonyms for the above, but with the STIX fonts we want to use the alternate alphabet.

```

226 \@@_new_alphabet_config:nnn {cal} {Latin}
227 {
228   \@@_set_mathalphabet_Latin:nnn {cal}      {up,it} {#1}

```

```

229 \@@_set_mathalphabet_pos:nnnn {cal} {B} {up,it} {#1}
230 \@@_set_mathalphabet_pos:nnnn {cal} {E} {up,it} {#1}
231 \@@_set_mathalphabet_pos:nnnn {cal} {F} {up,it} {#1}
232 \@@_set_mathalphabet_pos:nnnn {cal} {H} {up,it} {#1}
233 \@@_set_mathalphabet_pos:nnnn {cal} {I} {up,it} {#1}
234 \@@_set_mathalphabet_pos:nnnn {cal} {L} {up,it} {#1}
235 \@@_set_mathalphabet_pos:nnnn {cal} {M} {up,it} {#1}
236 \@@_set_mathalphabet_pos:nnnn {cal} {R} {up,it} {#1}
237 }
238 \@@_new_alphabet_config:nnn {cal} {latin}
239 {
240 \@@_set_mathalphabet_latin:nnn {cal} {up,it} {#1}
241 \@@_set_mathalphabet_pos:nnnn {cal} {e} {up,it} {#1}
242 \@@_set_mathalphabet_pos:nnnn {cal} {g} {up,it} {#1}
243 \@@_set_mathalphabet_pos:nnnn {cal} {o} {up,it} {#1}
244 }

```

17.5 *Fraktur or fraktur or blackletter: frak*

```

245 \@@_new_alphabet_config:nnn {frak} {Latin}
246 {
247 \@@_set_mathalphabet_Latin:nnn {frak} {up,it} {#1}
248 \@@_set_mathalphabet_pos:nnnn {frak} {C} {up,it} {#1}
249 \@@_set_mathalphabet_pos:nnnn {frak} {H} {up,it} {#1}
250 \@@_set_mathalphabet_pos:nnnn {frak} {I} {up,it} {#1}
251 \@@_set_mathalphabet_pos:nnnn {frak} {R} {up,it} {#1}
252 \@@_set_mathalphabet_pos:nnnn {frak} {Z} {up,it} {#1}
253 }
254 \@@_new_alphabet_config:nnn {frak} {latin}
255 {
256 \@@_set_mathalphabet_latin:nnn {frak} {up,it} {#1}
257 }

```

17.6 *Sans serif upright: sfup*

```

258 \@@_new_alphabet_config:nnn {sfup} {num}
259 {
260 \@@_set_mathalphabet_numbers:nnn {sf} {up} {#1}
261 \@@_set_mathalphabet_numbers:nnn {sfup} {up} {#1}
262 }
263 \@@_new_alphabet_config:nnn {sfup} {Latin}
264 {
265 \bool_if:NTF \g_@@_sfliteral_bool
266 {
267 \@@_set_normal_Latin:nn {sfup} {#1}
268 \@@_set_mathalphabet_Latin:nnn {sf} {up} {#1}
269 }
270 {
271 \bool_if:NT \g_@@_upsans_bool
272 {
273 \@@_set_normal_Latin:nn {sfup,sfit} {#1}

```

```

274         \@@_set_mathalphabet_Latin:nnn {sf} {up,it} {#1}
275     }
276 }
277 \@@_set_mathalphabet_Latin:nnn {sfup} {up,it} {#1}
278 }
279
280 \@@_new_alphabet_config:nnn {sfup} {latin}
281 {
282     \bool_if:NTF \g_@@_sfliteral_bool
283     {
284         \@@_set_normal_latin:nn {sfup} {#1}
285         \@@_set_mathalphabet_latin:nnn {sf} {up} {#1}
286     }
287     {
288         \bool_if:NT \g_@@_upsans_bool
289         {
290             \@@_set_normal_latin:nn {sfup,sfit} {#1}
291             \@@_set_mathalphabet_latin:nnn {sf} {up,it} {#1}
292         }
293     }
294     \@@_set_mathalphabet_latin:nnn {sfup} {up,it} {#1}
295 }

```

17.7 *Sans serif italic: sfit*

```

296 \@@_new_alphabet_config:nnn {sfit} {Latin}
297 {
298     \bool_if:NTF \g_@@_sfliteral_bool
299     {
300         \@@_set_normal_Latin:nn {sfit} {#1}
301         \@@_set_mathalphabet_Latin:nnn {sf} {it} {#1}
302     }
303     {
304         \bool_if:NF \g_@@_upsans_bool
305         {
306             \@@_set_normal_Latin:nn {sfup,sfit} {#1}
307             \@@_set_mathalphabet_Latin:nnn {sf} {up,it} {#1}
308         }
309     }
310     \@@_set_mathalphabet_Latin:nnn {sfit} {up,it} {#1}
311 }
312
313 \@@_new_alphabet_config:nnn {sfit} {latin}
314 {
315     \bool_if:NTF \g_@@_sfliteral_bool
316     {
317         \@@_set_normal_latin:nn {sfit} {#1}
318         \@@_set_mathalphabet_latin:nnn {sf} {it}{#1}
319     }
320     {

```

```

321     \bool_if:NF \g_@@_upsans_bool
322     {
323         \@@_set_normal_latin:nn {sfup,sfit} {#1}
324         \@@_set_mathalphabet_latin:nnn {sf} {up,it}{#1}
325     }
326 }
327 \@@_set_mathalphabet_latin:nnn {sfit} {up,it}{#1}
328 }

```

17.8 *Typewriter or monospaced: tt*

```

329 \@@_new_alphabet_config:nnn {tt} {num}
330 {
331     \@@_set_mathalphabet_numbers:nnn {tt} {up}{#1}
332 }
333 \@@_new_alphabet_config:nnn {tt} {Latin}
334 {
335     \@@_set_mathalphabet_Latin:nnn {tt} {up,it}{#1}
336 }
337 \@@_new_alphabet_config:nnn {tt} {latin}
338 {
339     \@@_set_mathalphabet_latin:nnn {tt} {up,it}{#1}
340 }

```

17.9 *Bold Italic: bfit*

```

341 \@@_new_alphabet_config:nnn {bfit} {Latin}
342 {
343     \bool_if:NF \g_@@_bfupLatin_bool
344     {
345         \@@_set_normal_Latin:nn {bfup,bfit} {#1}
346     }
347     \@@_set_mathalphabet_Latin:nnn {bfit} {up,it}{#1}
348     \bool_if:NTF \g_@@_bfliteral_bool
349     {
350         \@@_set_normal_Latin:nn {bfit} {#1}
351         \@@_set_mathalphabet_Latin:nnn {bf} {it}{#1}
352     }
353     {
354         \bool_if:NF \g_@@_bfupLatin_bool
355         {
356             \@@_set_normal_Latin:nn {bfup,bfit} {#1}
357             \@@_set_mathalphabet_Latin:nnn {bf} {up,it}{#1}
358         }
359     }
360 }
361
362 \@@_new_alphabet_config:nnn {bfit} {latin}
363 {
364     \bool_if:NF \g_@@_bfuplatin_bool
365     {

```

```

366     \@@_set_normal_latin:nn {bfup,bfit} {#1}
367   }
368   \@@_set_mathalphabet_latin:nnn {bfit} {up,it}{#1}
369   \bool_if:NTF \g_@@_bfliteral_bool
370   {
371     \@@_set_normal_latin:nn {bfit} {#1}
372     \@@_set_mathalphabet_latin:nnn {bf} {it}{#1}
373   }
374   {
375     \bool_if:NF \g_@@_bfuplatin_bool
376     {
377       \@@_set_normal_latin:nn {bfup,bfit} {#1}
378       \@@_set_mathalphabet_latin:nnn {bf} {up,it}{#1}
379     }
380   }
381 }
382
383 \@@_new_alphabet_config:nnn {bfit} {Greek}
384 {
385   \@@_set_mathalphabet_Greek:nnn {bfit} {up,it}{#1}
386   \bool_if:NTF \g_@@_bfliteral_bool
387   {
388     \@@_set_normal_Greek:nn {bfit}{#1}
389     \@@_set_mathalphabet_Greek:nnn {bf} {it}{#1}
390   }
391   {
392     \bool_if:NF \g_@@_bfupGreek_bool
393     {
394       \@@_set_normal_Greek:nn {bfup,bfit}{#1}
395       \@@_set_mathalphabet_Greek:nnn {bf} {up,it}{#1}
396     }
397   }
398 }
399
400 \@@_new_alphabet_config:nnn {bfit} {greek}
401 {
402   \@@_set_mathalphabet_greek:nnn {bfit} {up,it} {#1}
403   \bool_if:NTF \g_@@_bfliteral_bool
404   {
405     \@@_set_normal_greek:nn {bfit} {#1}
406     \@@_set_mathalphabet_greek:nnn {bf} {it} {#1}
407   }
408   {
409     \bool_if:NF \g_@@_bfupgreek_bool
410     {
411       \@@_set_normal_greek:nn {bfit,bfup} {#1}
412       \@@_set_mathalphabet_greek:nnn {bf} {up,it} {#1}
413     }
414   }

```



```

415 }
416
417 \@@_new_alphabet_config:nnn {bfit} {misc}
418 {
419   \bool_if:NTF \g_@@_literal_Nabla_bool
420     { \@@_set_normal_char:nnn {Nabla} {bfit} {#1} }
421     {
422       \bool_if:NF \g_@@_upNabla_bool
423         { \@@_set_normal_char:nnn {Nabla} {bfup,bfit} {#1} }
424     }
425
426   \bool_if:NTF \g_@@_literal_partial_bool
427     { \@@_set_normal_char:nnn {partial} {bfit} {#1} }
428     {
429       \bool_if:NF \g_@@_uppartial_bool
430         { \@@_set_normal_char:nnn {partial} {bfup,bfit} {#1} }
431     }
432
433   \@@_set_mathalphabet_pos:nnnn {bfit} {partial} {up,it} {#1}
434   \@@_set_mathalphabet_pos:nnnn {bfit} {Nabla} {up,it} {#1}
435
436   \bool_if:NTF \g_@@_literal_partial_bool
437     {
438       \@@_set_mathalphabet_pos:nnnn {bf} {partial} {it}{#1}
439     }
440     {
441       \bool_if:NF \g_@@_uppartial_bool
442         {
443           \@@_set_mathalphabet_pos:nnnn {bf} {partial} {up,it}{#1}
444         }
445     }
446
447   \bool_if:NTF \g_@@_literal_Nabla_bool
448     {
449       \@@_set_mathalphabet_pos:nnnn {bf} {Nabla} {it}{#1}
450     }
451     {
452       \bool_if:NF \g_@@_upNabla_bool
453         {
454           \@@_set_mathalphabet_pos:nnnn {bf} {Nabla} {up,it}{#1}
455         }
456     }
457 }

```

17.10 Bold Upright: *bfup*

```

458 \@@_new_alphabet_config:nnn {bfup} {num}
459 {
460   \@@_set_mathalphabet_numbers:nnn {bf} {up} {#1}
461   \@@_set_mathalphabet_numbers:nnn {bfup} {up} {#1}

```

```

462 }
463
464 \@@_new_alphabet_config:nnn {bfup} {Latin}
465 {
466   \bool_if:NT \g_@@_bfupLatin_bool
467   {
468     \@@_set_normal_Latin:nn {bfup,bfit} {#1}
469   }
470   \@@_set_mathalphabet_Latin:nnn {bfup} {up,it} {#1}
471   \bool_if:NTF \g_@@_bfliteral_bool
472   {
473     \@@_set_normal_Latin:nn {bfup} {#1}
474     \@@_set_mathalphabet_Latin:nnn {bf} {up} {#1}
475   }
476   {
477     \bool_if:NT \g_@@_bfupLatin_bool
478     {
479       \@@_set_normal_Latin:nn {bfup,bfit} {#1}
480       \@@_set_mathalphabet_Latin:nnn {bf} {up,it} {#1}
481     }
482   }
483 }
484
485 \@@_new_alphabet_config:nnn {bfup} {latin}
486 {
487   \bool_if:NT \g_@@_bfuplatin_bool
488   {
489     \@@_set_normal_latin:nn {bfup,bfit} {#1}
490   }
491   \@@_set_mathalphabet_latin:nnn {bfup} {up,it} {#1}
492   \bool_if:NTF \g_@@_bfliteral_bool
493   {
494     \@@_set_normal_latin:nn {bfup} {#1}
495     \@@_set_mathalphabet_latin:nnn {bf} {up} {#1}
496   }
497   {
498     \bool_if:NT \g_@@_bfuplatin_bool
499     {
500       \@@_set_normal_latin:nn {bfup,bfit} {#1}
501       \@@_set_mathalphabet_latin:nnn {bf} {up,it} {#1}
502     }
503   }
504 }
505
506 \@@_new_alphabet_config:nnn {bfup} {Greek}
507 {
508   \@@_set_mathalphabet_Greek:nnn {bfup} {up,it} {#1}
509   \bool_if:NTF \g_@@_bfliteral_bool
510   {

```

```

511     \@@_set_normal_Greek:nn {bfup} {#1}
512     \@@_set_mathalphabet_Greek:nnn {bf} {up} {#1}
513   }
514   {
515     \bool_if:NT \g_@@_bfupGreek_bool
516     {
517       \@@_set_normal_Greek:nn {bfup,bfit} {#1}
518       \@@_set_mathalphabet_Greek:nnn {bf} {up,it} {#1}
519     }
520   }
521 }
522
523 \@@_new_alphabet_config:nnn {bfup} {greek}
524 {
525   \@@_set_mathalphabet_greek:nnn {bfup} {up,it} {#1}
526   \bool_if:NTF \g_@@_bfliteral_bool
527   {
528     \@@_set_normal_greek:nn {bfup} {#1}
529     \@@_set_mathalphabet_greek:nnn {bf} {up} {#1}
530   }
531   {
532     \bool_if:NT \g_@@_bfupgreek_bool
533     {
534       \@@_set_normal_greek:nn {bfup,bfit} {#1}
535       \@@_set_mathalphabet_greek:nnn {bf} {up,it} {#1}
536     }
537   }
538 }
539
540 \@@_new_alphabet_config:nnn {bfup} {misc}
541 {
542   \bool_if:NTF \g_@@_literal_Nabla_bool
543   {
544     \@@_set_normal_char:nnn {Nabla} {bfup} {#1}
545   }
546   {
547     \bool_if:NT \g_@@_upNabla_bool
548     {
549       \@@_set_normal_char:nnn {Nabla} {bfup,bfit} {#1}
550     }
551   }
552   \bool_if:NTF \g_@@_literal_partial_bool
553   {
554     \@@_set_normal_char:nnn {partial} {bfup} {#1}
555   }
556   {
557     \bool_if:NT \g_@@_uppartial_bool
558     {
559       \@@_set_normal_char:nnn {partial} {bfup,bfit} {#1}

```

```

560     }
561   }
562   \@@_set_mathalphabet_pos:nnnn {bfup} {partial} {up,it} {#1}
563   \@@_set_mathalphabet_pos:nnnn {bfup} {Nabla} {up,it} {#1}
564   \@@_set_mathalphabet_pos:nnnn {bfup} {digamma} {up} {#1}
565   \@@_set_mathalphabet_pos:nnnn {bfup} {Digamma} {up} {#1}
566   \@@_set_mathalphabet_pos:nnnn {bf} {digamma} {up} {#1}
567   \@@_set_mathalphabet_pos:nnnn {bf} {Digamma} {up} {#1}
568   \bool_if:NTF \g_@@_literal_partial_bool
569   {
570     \@@_set_mathalphabet_pos:nnnn {bf} {partial} {up} {#1}
571   }
572   {
573     \bool_if:NT \g_@@_uppartial_bool
574     {
575       \@@_set_mathalphabet_pos:nnnn {bf} {partial} {up,it} {#1}
576     }
577   }
578   \bool_if:NTF \g_@@_literal_Nabla_bool
579   {
580     \@@_set_mathalphabet_pos:nnnn {bf} {Nabla} {up}{#1}
581   }
582   {
583     \bool_if:NT \g_@@_upNabla_bool
584     {
585       \@@_set_mathalphabet_pos:nnnn {bf} {Nabla} {up,it} {#1}
586     }
587   }
588 }

```

17.11 *Bold fractur or fraktur or blackletter: bffrak*

```

589 \@@_new_alphabet_config:nnn {bffrak} {Latin}
590 {
591   \@@_set_mathalphabet_Latin:nnn {bffrak} {up,it}{#1}
592 }
593
594 \@@_new_alphabet_config:nnn {bffrak} {latin}
595 {
596   \@@_set_mathalphabet_latin:nnn {bffrak} {up,it}{#1}
597 }

```

17.12 *Bold script or calligraphic: bfscr*

```

598 \@@_new_alphabet_config:nnn {bfscr} {Latin}
599 {
600   \@@_set_mathalphabet_Latin:nnn {bfscr} {up,it}{#1}
601 }
602 \@@_new_alphabet_config:nnn {bfscr} {latin}
603 {
604   \@@_set_mathalphabet_latin:nnn {bfscr} {up,it}{#1}

```

```

605 }
606 \@@_new_alphabet_config:nnn {bfc} {Latin}
607 {
608   \@@_set_mathalphabet_Latin:nnn {bfc} {up,it}{#1}
609 }
610 \@@_new_alphabet_config:nnn {bfc} {latin}
611 {
612   \@@_set_mathalphabet_latin:nnn {bfc} {up,it}{#1}
613 }

```

17.13 Bold upright sans serif: *bfsfup*

```

614 \@@_new_alphabet_config:nnn {bfsfup} {num}
615 {
616   \@@_set_mathalphabet_numbers:nnn {bfsf} {up}{#1}
617   \@@_set_mathalphabet_numbers:nnn {bfsfup} {up}{#1}
618 }
619 \@@_new_alphabet_config:nnn {bfsfup} {Latin}
620 {
621   \bool_if:NTF \g_@@_sfliteral_bool
622   {
623     \@@_set_normal_Latin:nn {bfsfup} {#1}
624     \@@_set_mathalphabet_Latin:nnn {bfsf} {up}{#1}
625   }
626   {
627     \bool_if:NT \g_@@_upsans_bool
628     {
629       \@@_set_normal_Latin:nn {bfsfup,bfsfit} {#1}
630       \@@_set_mathalphabet_Latin:nnn {bfsf} {up,it}{#1}
631     }
632   }
633   \@@_set_mathalphabet_Latin:nnn {bfsfup} {up,it}{#1}
634 }
635
636 \@@_new_alphabet_config:nnn {bfsfup} {latin}
637 {
638   \bool_if:NTF \g_@@_sfliteral_bool
639   {
640     \@@_set_normal_latin:nn {bfsfup} {#1}
641     \@@_set_mathalphabet_latin:nnn {bfsf} {up}{#1}
642   }
643   {
644     \bool_if:NT \g_@@_upsans_bool
645     {
646       \@@_set_normal_latin:nn {bfsfup,bfsfit} {#1}
647       \@@_set_mathalphabet_latin:nnn {bfsf} {up,it}{#1}
648     }
649   }
650   \@@_set_mathalphabet_latin:nnn {bfsfup} {up,it}{#1}
651 }

```

```

652
653 \@@_new_alphabet_config:nnn {bfsfup} {Greek}
654 {
655   \bool_if:NTF \g_@@_sfliteral_bool
656   {
657     \@@_set_normal_Greek:nn {bfsfup}{#1}
658     \@@_set_mathalphabet_Greek:nnn {bfsf} {up}{#1}
659   }
660   {
661     \bool_if:NT \g_@@_upsans_bool
662     {
663       \@@_set_normal_Greek:nn {bfsfup,bfsfit}{#1}
664       \@@_set_mathalphabet_Greek:nnn {bfsf} {up,it}{#1}
665     }
666   }
667   \@@_set_mathalphabet_Greek:nnn {bfsfup} {up,it}{#1}
668 }
669
670 \@@_new_alphabet_config:nnn {bfsfup} {greek}
671 {
672   \bool_if:NTF \g_@@_sfliteral_bool
673   {
674     \@@_set_normal_greek:nn {bfsfup} {#1}
675     \@@_set_mathalphabet_greek:nnn {bfsf} {up} {#1}
676   }
677   {
678     \bool_if:NT \g_@@_upsans_bool
679     {
680       \@@_set_normal_greek:nn {bfsfup,bfsfit} {#1}
681       \@@_set_mathalphabet_greek:nnn {bfsf} {up,it} {#1}
682     }
683   }
684   \@@_set_mathalphabet_greek:nnn {bfsfup} {up,it} {#1}
685 }
686
687 \@@_new_alphabet_config:nnn {bfsfup} {misc}
688 {
689   \bool_if:NTF \g_@@_literal_Nabla_bool
690   {
691     \@@_set_normal_char:nnn {Nabla}{bfsfup}{#1}
692   }
693   {
694     \bool_if:NT \g_@@_upNabla_bool
695     {
696       \@@_set_normal_char:nnn {Nabla}{bfsfup,bfsfit}{#1}
697     }
698   }
699   \bool_if:NTF \g_@@_literal_partial_bool
700   {

```

```

701 \@@_set_normal_char:nnn {partial}{bfsfup}{#1}
702 }
703 {
704 \bool_if:NT \g_@@_uppartial_bool
705 {
706 \@@_set_normal_char:nnn {partial}{bfsfup,bfsfit}{#1}
707 }
708 }
709 \@@_set_mathalphabet_pos:nxxx {bfsfup} {partial} {up,it}{#1}
710 \@@_set_mathalphabet_pos:nxxx {bfsfup} {Nabla} {up,it}{#1}
711 \bool_if:NTF \g_@@_literal_partial_bool
712 {
713 \@@_set_mathalphabet_pos:nxxx {bfsf} {partial} {up}{#1}
714 }
715 {
716 \bool_if:NT \g_@@_uppartial_bool
717 {
718 \@@_set_mathalphabet_pos:nxxx {bfsf} {partial} {up,it}{#1}
719 }
720 }
721 \bool_if:NTF \g_@@_literal_Nabla_bool
722 {
723 \@@_set_mathalphabet_pos:nxxx {bfsf} {Nabla} {up}{#1}
724 }
725 {
726 \bool_if:NT \g_@@_upNabla_bool
727 {
728 \@@_set_mathalphabet_pos:nxxx {bfsf} {Nabla} {up,it}{#1}
729 }
730 }
731 }

```

17.14 *Bold italic sans serif: bfsfit*

```

732 \@@_new_alphabet_config:nnn {bfsfit} {Latin}
733 {
734 \bool_if:NTF \g_@@_sfliteral_bool
735 {
736 \@@_set_normal_Latin:nn {bfsfit} {#1}
737 \@@_set_mathalphabet_Latin:nnn {bfsf} {it}{#1}
738 }
739 {
740 \bool_if:NF \g_@@_upsans_bool
741 {
742 \@@_set_normal_Latin:nn {bfsfup,bfsfit} {#1}
743 \@@_set_mathalphabet_Latin:nnn {bfsf} {up,it}{#1}
744 }
745 }
746 \@@_set_mathalphabet_Latin:nnn {bfsfit} {up,it}{#1}
747 }

```

```

748
749 \@@_new_alphabet_config:nnn {bfsfit} {latin}
750 {
751   \bool_if:NTF \g_@@_sfliteral_bool
752   {
753     \@@_set_normal_latin:nn {bfsfit} {#1}
754     \@@_set_mathalphabet_latin:nnn {bfsf} {it}{#1}
755   }
756   {
757     \bool_if:NF \g_@@_upsans_bool
758     {
759       \@@_set_normal_latin:nn {bfsfup,bfsfit} {#1}
760       \@@_set_mathalphabet_latin:nnn {bfsf} {up,it}{#1}
761     }
762   }
763   \@@_set_mathalphabet_latin:nnn {bfsfit} {up,it}{#1}
764 }
765
766 \@@_new_alphabet_config:nnn {bfsfit} {Greek}
767 {
768   \bool_if:NTF \g_@@_sfliteral_bool
769   {
770     \@@_set_normal_Greek:nn {bfsfit}{#1}
771     \@@_set_mathalphabet_Greek:nnn {bfsf} {it}{#1}
772   }
773   {
774     \bool_if:NF \g_@@_upsans_bool
775     {
776       \@@_set_normal_Greek:nn {bfsfup,bfsfit}{#1}
777       \@@_set_mathalphabet_Greek:nnn {bfsf} {up,it}{#1}
778     }
779   }
780   \@@_set_mathalphabet_Greek:nnn {bfsfit} {up,it}{#1}
781 }
782
783 \@@_new_alphabet_config:nnn {bfsfit} {greek}
784 {
785   \bool_if:NTF \g_@@_sfliteral_bool
786   {
787     \@@_set_normal_greek:nn {bfsfit} {#1}
788     \@@_set_mathalphabet_greek:nnn {bfsf} {it} {#1}
789   }
790   {
791     \bool_if:NF \g_@@_upsans_bool
792     {
793       \@@_set_normal_greek:nn {bfsfup,bfsfit} {#1}
794       \@@_set_mathalphabet_greek:nnn {bfsf} {up,it} {#1}
795     }
796   }

```



```

797 \@@_set_mathalphabet_greek:nnn {bfsfit} {up,it} {#1}
798 }
799
800 \@@_new_alphabet_config:nnn {bfsfit} {misc}
801 {
802 \bool_if:NTF \g_@@_literal_Nabla_bool
803 {
804 \@@_set_normal_char:nnn {Nabla}{bfsfit}{#1}
805 }
806 {
807 \bool_if:NF \g_@@_upNabla_bool
808 {
809 \@@_set_normal_char:nnn {Nabla}{bfsfup,bfsfit}{#1}
810 }
811 }
812 \bool_if:NTF \g_@@_literal_partial_bool
813 {
814 \@@_set_normal_char:nnn {partial}{bfsfit}{#1}
815 }
816 {
817 \bool_if:NF \g_@@_uppartial_bool
818 {
819 \@@_set_normal_char:nnn {partial}{bfsfup,bfsfit}{#1}
820 }
821 }
822 \@@_set_mathalphabet_pos:nnnn {bfsfit} {partial} {up,it}{#1}
823 \@@_set_mathalphabet_pos:nnnn {bfsfit} {Nabla} {up,it}{#1}
824 \bool_if:NTF \g_@@_literal_partial_bool
825 {
826 \@@_set_mathalphabet_pos:nnnn {bfsf} {partial} {it}{#1}
827 }
828 {
829 \bool_if:NF \g_@@_uppartial_bool
830 {
831 \@@_set_mathalphabet_pos:nnnn {bfsf} {partial} {up,it}{#1}
832 }
833 }
834 \bool_if:NTF \g_@@_literal_Nabla_bool
835 {
836 \@@_set_mathalphabet_pos:nnnn {bfsf} {Nabla} {it}{#1}
837 }
838 {
839 \bool_if:NF \g_@@_upNabla_bool
840 {
841 \@@_set_mathalphabet_pos:nnnn {bfsf} {Nabla} {up,it}{#1}
842 }
843 }
844 }
845 (/package)

```

File XVII

um-code-primes.dtx

18 Primes

1 `(*package)`

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

u+2032 prime (`\prime`): x'

u+2033 double prime (`\dprime`): x''

u+2034 triple prime (`\trprime`): x'''

u+2057 quadruple prime (`\qprime`): x''''

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

x' x'' x''' x''''

The glyphs are now ‘full size’ so that when placed inside a superscript, their shape will match the originally sized ones. Many thanks to Ross Mills of Tiro Typeworks for originally pointing out this behaviour.

In regular \LaTeX , primes can be entered with the straight quote character `'`, and multiple straight quotes chain together to produce multiple primes. Better results can be achieved in `unicode-math` by chaining multiple single primes into a pre-drawn multi-prime glyph; consider x''' vs. x''' .

For Unicode maths, we wish to conserve this behaviour and augment it with the possibility of adding any combination of Unicode prime or any of the n -prime characters. E.g., the user might copy-paste a double prime from another source and then later type another single prime after it; the output should be the triple prime.

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

This is a wrapper to insert a superscript; if there is a subsequent trailing superscript, then it is included within the insertion.

2 `\cs_new:Nn \@@_arg_i_before_egroup:n {#1\egroup}`

```

3 \cs_new:Nn \@@_superscript:n
4 {
5   ^\bgroup #1
6   \peek_meaning_remove:NTF ^ \@@_arg_i_before_egroup:n \egroup
7 }
8 \cs_new:Nn \@@_nprimes:Nn
9 {
10  \@@_superscript:n
11  {
12    #1
13    \prg_replicate:nn {#2-1} { \mskip \g_@@_primekern_muskip #1 }
14  }
15 }
16 \cs_new:Nn \@@_nprimes_select:nn
17 {
18  \int_case:nnF {#2}
19  {
20    {1} { \@@_superscript:n {#1} }
21    {2} {
22      \@@_glyph_if_exist:NnTF \g_@@_prime_font_cmd_tl {"2033}
23      { \@@_superscript:n {\@@_prime_double_mchar} }
24      { \@@_nprimes:Nn #1 {#2} }
25    }
26    {3} {
27      \@@_glyph_if_exist:NnTF \g_@@_prime_font_cmd_tl {"2034}
28      { \@@_superscript:n {\@@_prime_triple_mchar} }
29      { \@@_nprimes:Nn #1 {#2} }
30    }
31    {4} {
32      \@@_glyph_if_exist:NnTF \g_@@_prime_font_cmd_tl {"2057}
33      { \@@_superscript:n {\@@_prime_quad_mchar} }
34      { \@@_nprimes:Nn #1 {#2} }
35    }
36  }
37  {
38    \@@_nprimes:Nn #1 {#2}
39  }
40 }
41 \cs_new:Nn \@@_nbackprimes_select:nn
42 {
43  \int_case:nnF {#2}
44  {
45    {1} { \@@_superscript:n {#1} }
46    {2} {
47      \@@_glyph_if_exist:NnTF \g_@@_prime_font_cmd_tl {"2036}
48      { \@@_superscript:n {\@@_backprime_double_mchar} }
49      { \@@_nprimes:Nn #1 {#2} }
50    }

```

```

51 {3} {
52   \@@_glyph_if_exist:NnTF \g_@@_prime_font_cmd_tl {"2037}
53   { \@@_superscript:n {\@@_backprime_triple_mchar} }
54   { \@@_nprimes:Nn #1 {#2} }
55 }
56 }
57 {
58   \@@_nprimes:Nn #1 {#2}
59 }
60 }

```

Scanning is annoying because I'm too lazy to do it for the general case.

```

61 \cs_new:Npn \@@_scan_prime:
62 {
63   \cs_set_eq:NN \@@_superscript:n \use:n
64   \int_zero:N \l_@@_primecount_int
65   \@@_scanprime_collect:N \@@_prime_single_mchar
66 }
67 \cs_new:Npn \@@_scan_dprime:
68 {
69   \cs_set_eq:NN \@@_superscript:n \use:n
70   \int_set:Nn \l_@@_primecount_int {1}
71   \@@_scanprime_collect:N \@@_prime_single_mchar
72 }
73 \cs_new:Npn \@@_scan_trprime:
74 {
75   \cs_set_eq:NN \@@_superscript:n \use:n
76   \int_set:Nn \l_@@_primecount_int {2}
77   \@@_scanprime_collect:N \@@_prime_single_mchar
78 }
79 \cs_new:Npn \@@_scan_qprime:
80 {
81   \cs_set_eq:NN \@@_superscript:n \use:n
82   \int_set:Nn \l_@@_primecount_int {3}
83   \@@_scanprime_collect:N \@@_prime_single_mchar
84 }
85 \cs_new:Npn \@@_scan_sup_prime:
86 {
87   \int_zero:N \l_@@_primecount_int
88   \@@_scanprime_collect:N \@@_prime_single_mchar
89 }
90 \cs_new:Npn \@@_scan_sup_dprime:
91 {
92   \int_set:Nn \l_@@_primecount_int {1}
93   \@@_scanprime_collect:N \@@_prime_single_mchar
94 }
95 \cs_new:Npn \@@_scan_sup_trprime:
96 {
97   \int_set:Nn \l_@@_primecount_int {2}
98   \@@_scanprime_collect:N \@@_prime_single_mchar

```

```

99 }
100 \cs_new:Npn \@@_scan_sup_qprime:
101 {
102   \int_set:Nn \l_@@_primecount_int {3}
103   \@@_scanprime_collect:N \@@_prime_single_mchar
104 }
105 \cs_new:Nn \@@_scanprime_collect:N
106 {
107   \int_incr:N \l_@@_primecount_int
108   \peek_meaning_remove:NTF '
109   { \@@_scanprime_collect:N #1 }
110   {
111     \peek_meaning_remove:NTF \@@_scan_prime:
112     { \@@_scanprime_collect:N #1 }
113     {
114       \peek_meaning_remove:NTF ^^^^2032
115       { \@@_scanprime_collect:N #1 }
116       {
117         \peek_meaning_remove:NTF \@@_scan_dprime:
118         {
119           \int_incr:N \l_@@_primecount_int
120           \@@_scanprime_collect:N #1
121         }
122         {
123           \peek_meaning_remove:NTF ^^^^2033
124           {
125             \int_incr:N \l_@@_primecount_int
126             \@@_scanprime_collect:N #1
127           }
128           {
129             \peek_meaning_remove:NTF \@@_scan_trprime:
130             {
131               \int_add:Nn \l_@@_primecount_int {2}
132               \@@_scanprime_collect:N #1
133             }
134             {
135               \peek_meaning_remove:NTF ^^^^2034
136               {
137                 \int_add:Nn \l_@@_primecount_int {2}
138                 \@@_scanprime_collect:N #1
139               }
140               {
141                 \peek_meaning_remove:NTF \@@_scan_qprime:
142                 {
143                   \int_add:Nn \l_@@_primecount_int {3}
144                   \@@_scanprime_collect:N #1
145                 }
146                 {
147                   \peek_meaning_remove:NTF ^^^^2057

```

```

148         {
149             \int_add:Nn \l_@@_primecount_int {3}
150             \@@_scanprime_collect:N #1
151         }
152         {
153             \@@_nprimes_select:nn {#1} {\l_@@_primecount_int}
154         }
155     }
156 }
157 }
158 }
159 }
160 }
161 }
162 }
163 }

164 \cs_new:Npn \@@_scan_backprime:
165 {
166     \cs_set_eq:NN \@@_superscript:n \use:n
167     \int_zero:N \l_@@_primecount_int
168     \@@_scanbackprime_collect:N \@@_backprime_single_mchar
169 }

170 \cs_new:Npn \@@_scan_backdprime:
171 {
172     \cs_set_eq:NN \@@_superscript:n \use:n
173     \int_set:Nn \l_@@_primecount_int {1}
174     \@@_scanbackprime_collect:N \@@_backprime_single_mchar
175 }

176 \cs_new:Npn \@@_scan_backtrprime:
177 {
178     \cs_set_eq:NN \@@_superscript:n \use:n
179     \int_set:Nn \l_@@_primecount_int {2}
180     \@@_scanbackprime_collect:N \@@_backprime_single_mchar
181 }

182 \cs_new:Npn \@@_scan_sup_backprime:
183 {
184     \int_zero:N \l_@@_primecount_int
185     \@@_scanbackprime_collect:N \@@_backprime_single_mchar
186 }

187 \cs_new:Npn \@@_scan_sup_backdprime:
188 {
189     \int_set:Nn \l_@@_primecount_int {1}
190     \@@_scanbackprime_collect:N \@@_backprime_single_mchar
191 }

192 \cs_new:Npn \@@_scan_sup_backtrprime:
193 {
194     \int_set:Nn \l_@@_primecount_int {2}
195     \@@_scanbackprime_collect:N \@@_backprime_single_mchar
196 }

```

```

197 \cs_new:Nn \@@_scanbackprime_collect:N
198 {
199   \int_incr:N \l_@@_primecount_int
200   \peek_meaning_remove:NTF `
201   {
202     \@@_scanbackprime_collect:N #1
203   }
204   {
205     \peek_meaning_remove:NTF \@@_scan_backprime:
206     {
207       \@@_scanbackprime_collect:N #1
208     }
209     {
210       \peek_meaning_remove:NTF ^^^^2035
211       {
212         \@@_scanbackprime_collect:N #1
213       }
214       {
215         \peek_meaning_remove:NTF \@@_scan_backdprime:
216         {
217           \int_incr:N \l_@@_primecount_int
218           \@@_scanbackprime_collect:N #1
219         }
220         {
221           \peek_meaning_remove:NTF ^^^^2036
222           {
223             \int_incr:N \l_@@_primecount_int
224             \@@_scanbackprime_collect:N #1
225           }
226           {
227             \peek_meaning_remove:NTF \@@_scan_backtrprime:
228             {
229               \int_add:Nn \l_@@_primecount_int {2}
230               \@@_scanbackprime_collect:N #1
231             }
232             {
233               \peek_meaning_remove:NTF ^^^^2037
234               {
235                 \int_add:Nn \l_@@_primecount_int {2}
236                 \@@_scanbackprime_collect:N #1
237               }
238               {
239                 \@@_nbackprimes_select:nn {#1} {\l_@@_primecount_int}
240               }
241             }
242           }
243         }
244       }
245     }

```

```

246     }
247 }

248 \AtBeginDocument { \@@_define_prime_commands: \@@_define_prime_chars: }
249 \cs_new:Nn \@@_define_prime_commands:
250 {
251   \cs_set_eq:NN \prime      \@@_prime_single_mchar
252   \cs_set_eq:NN \dprime    \@@_prime_double_mchar
253   \cs_set_eq:NN \trprime   \@@_prime_triple_mchar
254   \cs_set_eq:NN \qprime    \@@_prime_quad_mchar
255   \cs_set_eq:NN \backprime \@@_backprime_single_mchar
256   \cs_set_eq:NN \backdprime \@@_backprime_double_mchar
257   \cs_set_eq:NN \backtrprime \@@_backprime_triple_mchar
258 }

259 \group_begin:
260   \char_set_catcode_active:N \prime
261   \char_set_catcode_active:N \dprime
262   \char_set_catcode_active:n {"2032}
263   \char_set_catcode_active:n {"2033}
264   \char_set_catcode_active:n {"2034}
265   \char_set_catcode_active:n {"2057}
266   \char_set_catcode_active:n {"2035}
267   \char_set_catcode_active:n {"2036}
268   \char_set_catcode_active:n {"2037}
269   \cs_gset:Nn \@@_define_prime_chars:
270     {
271       \cs_set_eq:NN \prime      \@@_scan_sup_prime:
272       \cs_set_eq:NN \dprime    \@@_scan_sup_dprime:
273       \cs_set_eq:NN \trprime   \@@_scan_sup_trprime:
274       \cs_set_eq:NN \qprime    \@@_scan_sup_qprime:
275       \cs_set_eq:NN \backprime \@@_scan_sup_backprime:
276       \cs_set_eq:NN \backdprime \@@_scan_sup_backdprime:
277       \cs_set_eq:NN \backtrprime \@@_scan_sup_backtrprime:
278     }
279   \group_end:
280
281   \cs_set_eq:NN \active@math@prime \@@_scan_sup_prime:
282
283   \cs_set_eq:NN \active@math@prime \@@_scan_sup_prime:
284   </package>

```


File XVIII

um-code-sscript.dtx

19 Unicode sub- and super-scripts

```
1 (*package)
```

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?

Superscripts Populate a property list with superscript characters; themselves as their key, and their replacement as each key’s value. Then make the superscript active and bind it to the scanning function.

`\scantokens` makes this process much simpler since we can activate the char and assign its meaning in one step.

```
2 \cs_new:Nn \@@_setup_active_superscript:nn
3 {
4   \prop_gput:Nxn \g_@@_supers_prop { \int_eval:n {#1} } {#2}
5   \@@_mathactive_remap:nn {#1}
6   {
7     \tl_set:Nn \l_@@_ss_chain_tl {#2}
8     \cs_set_eq:NN \@@_sub_or_super:n \sp
9     \tl_set:Nn \l_@@_tmpa_tl {supers}
10    \@@_scan_sscript:
11  }
12 }
```

Subscripts

```
13 \cs_new:Nn \@@_setup_active_subscript:nn
14 {
15   \prop_gput:Nxn \g_@@_subs_prop { \int_eval:n {#1} } {#2}
16   \@@_mathactive_remap:nn {#1}
17   {
18     \tl_set:Nn \l_@@_ss_chain_tl {#2}
19     \cs_set_eq:NN \@@_sub_or_super:n \sb
20     \tl_set:Nn \l_@@_tmpa_tl {subs}
21     \@@_scan_sscript:
22   }
23 }
```

The scanning command Collects a chain of subscripts or a chain of superscripts and then typesets what it has collected.

```

24 \@@_cs_new:Nn \@@_scan_sscript:
25 {
26   \@@_scan_sscript:TF
27   { \@@_scan_sscript: }
28   { \@@_sub_or_super:n {\l_@@_ss_chain_tl} }
29 }

```

We do not skip spaces when scanning ahead, and we explicitly wish to bail out on encountering a space or a brace. These cases are filtered using `\peek_N_type:TF`. Otherwise the token can be taken as an N-type argument. Then we search for it in the appropriate property list (`\l_@@_tmpa_tl` is `subs` or `supers`). If found, add the value to the current chain of sub/superscripts. Remember to put the character back in the input otherwise. The `\group_align_safe_begin:` and `\group_align_safe_end:` are needed in case #3 is `&`.

```

30 \@@_cs_new:Nn \@@_scan_sscript:TF
31 {
32   \peek_N_type:TF
33   {
34     \group_align_safe_begin:
35     \@@_scan_sscript_aux:nnN {#1} {#2}
36   }
37   {#2}
38 }

```

The look-ahead for the sscripts doesn't try to peek inside the lookahead.

```

39 \@@_cs_new:Nn \@@_scan_sscript_aux:nnN
40 {
41   \tl_set:Nx \l_@@_tmpa_key_tl { \tl_to_str:n {#3} }
42   \prop_get:cxNTF {g_@@_l_@@_tmpa_tl _prop}
43   { \int_eval:n { \exp_after:wN ` \l_@@_tmpa_key_tl } }
44   \l_@@_tmpb_tl
45   {
46     \tl_put_right:NV \l_@@_ss_chain_tl \l_@@_tmpb_tl
47     \group_align_safe_end:
48     #1
49   }
50   { \group_align_safe_end: #2 #3 }
51 }

```

Definitions Superscripts.

```

52 \@@_setup_active_superscript:nn {"2070} {0}
53 \@@_setup_active_superscript:nn {"00B9} {1}
54 \@@_setup_active_superscript:nn {"00B2} {2}
55 \@@_setup_active_superscript:nn {"00B3} {3}
56 \@@_setup_active_superscript:nn {"2074} {4}
57 \@@_setup_active_superscript:nn {"2075} {5}
58 \@@_setup_active_superscript:nn {"2076} {6}

```

59 \@@_setup_active_superscript:nn {"2077} {7}
60 \@@_setup_active_superscript:nn {"2078} {8}
61 \@@_setup_active_superscript:nn {"2079} {9}
62 \@@_setup_active_superscript:nn {"207A} {+}
63 \@@_setup_active_superscript:nn {"207B} {-}
64 \@@_setup_active_superscript:nn {"207C} {=}
65 \@@_setup_active_superscript:nn {"207D} {(}
66 \@@_setup_active_superscript:nn {"207E} {)}
67 \@@_setup_active_superscript:nn {"1D2C} {A}
68 \@@_setup_active_superscript:nn {"1D2E} {B}
69 \@@_setup_active_superscript:nn {"1D30} {D}
70 \@@_setup_active_superscript:nn {"1D31} {E}
71 \@@_setup_active_superscript:nn {"1D33} {G}
72 \@@_setup_active_superscript:nn {"1D34} {H}
73 \@@_setup_active_superscript:nn {"1D35} {I}
74 \@@_setup_active_superscript:nn {"1D36} {J}
75 \@@_setup_active_superscript:nn {"1D37} {K}
76 \@@_setup_active_superscript:nn {"1D38} {L}
77 \@@_setup_active_superscript:nn {"1D39} {M}
78 \@@_setup_active_superscript:nn {"1D3A} {N}
79 \@@_setup_active_superscript:nn {"1D3C} {O}
80 \@@_setup_active_superscript:nn {"1D3E} {P}
81 \@@_setup_active_superscript:nn {"1D3F} {R}
82 \@@_setup_active_superscript:nn {"1D40} {T}
83 \@@_setup_active_superscript:nn {"1D41} {U}
84 \@@_setup_active_superscript:nn {"2C7D} {V}
85 \@@_setup_active_superscript:nn {"1D42} {W}
86 \@@_setup_active_superscript:nn {"1D43} {a}
87 \@@_setup_active_superscript:nn {"1D47} {b}
88 \@@_setup_active_superscript:nn {"1D9C} {c}
89 \@@_setup_active_superscript:nn {"1D48} {d}
90 \@@_setup_active_superscript:nn {"1D49} {e}
91 \@@_setup_active_superscript:nn {"1DA0} {f}
92 \@@_setup_active_superscript:nn {"1D4D} {g}
93 \@@_setup_active_superscript:nn {"02B0} {h}
94 \@@_setup_active_superscript:nn {"2071} {i}
95 \@@_setup_active_superscript:nn {"02B2} {j}
96 \@@_setup_active_superscript:nn {"1D4F} {k}
97 \@@_setup_active_superscript:nn {"02E1} {l}
98 \@@_setup_active_superscript:nn {"1D50} {m}
99 \@@_setup_active_superscript:nn {"207F} {n}
100 \@@_setup_active_superscript:nn {"1D52} {o}
101 \@@_setup_active_superscript:nn {"1D56} {p}
102 \@@_setup_active_superscript:nn {"02B3} {r}
103 \@@_setup_active_superscript:nn {"02E2} {s}
104 \@@_setup_active_superscript:nn {"1D57} {t}
105 \@@_setup_active_superscript:nn {"1D58} {u}
106 \@@_setup_active_superscript:nn {"1D5B} {v}
107 \@@_setup_active_superscript:nn {"02B7} {w}

```

108 \@@_setup_active_superscript:nn {"02E3} {x}
109 \@@_setup_active_superscript:nn {"02B8} {y}
110 \@@_setup_active_superscript:nn {"1DBB} {z}
111 \@@_setup_active_superscript:nn {"1D5D} {\beta}
112 \@@_setup_active_superscript:nn {"1D5E} {\gamma}
113 \@@_setup_active_superscript:nn {"1D5F} {\delta}
114 \@@_setup_active_superscript:nn {"1D60} {\phi}
115 \@@_setup_active_superscript:nn {"1D61} {\chi}
116 \@@_setup_active_superscript:nn {"1DBF} {\theta}

```

A few more subscripts than superscripts:

```

117 \@@_setup_active_subscript:nn {"2080} {0}
118 \@@_setup_active_subscript:nn {"2081} {1}
119 \@@_setup_active_subscript:nn {"2082} {2}
120 \@@_setup_active_subscript:nn {"2083} {3}
121 \@@_setup_active_subscript:nn {"2084} {4}
122 \@@_setup_active_subscript:nn {"2085} {5}
123 \@@_setup_active_subscript:nn {"2086} {6}
124 \@@_setup_active_subscript:nn {"2087} {7}
125 \@@_setup_active_subscript:nn {"2088} {8}
126 \@@_setup_active_subscript:nn {"2089} {9}
127 \@@_setup_active_subscript:nn {"208A} {+}
128 \@@_setup_active_subscript:nn {"208B} {-}
129 \@@_setup_active_subscript:nn {"208C} {=}
130 \@@_setup_active_subscript:nn {"208D} {(}
131 \@@_setup_active_subscript:nn {"208E} {)}
132 \@@_setup_active_subscript:nn {"2090} {a}
133 \@@_setup_active_subscript:nn {"2091} {e}
134 \@@_setup_active_subscript:nn {"2095} {h}
135 \@@_setup_active_subscript:nn {"1D62} {i}
136 \@@_setup_active_subscript:nn {"2C7C} {j}
137 \@@_setup_active_subscript:nn {"2096} {k}
138 \@@_setup_active_subscript:nn {"2097} {l}
139 \@@_setup_active_subscript:nn {"2098} {m}
140 \@@_setup_active_subscript:nn {"2099} {n}
141 \@@_setup_active_subscript:nn {"2092} {o}
142 \@@_setup_active_subscript:nn {"209A} {p}
143 \@@_setup_active_subscript:nn {"1D63} {r}
144 \@@_setup_active_subscript:nn {"209B} {s}
145 \@@_setup_active_subscript:nn {"209C} {t}
146 \@@_setup_active_subscript:nn {"1D64} {u}
147 \@@_setup_active_subscript:nn {"1D65} {v}
148 \@@_setup_active_subscript:nn {"2093} {x}
149 \@@_setup_active_subscript:nn {"1D66} {\beta}
150 \@@_setup_active_subscript:nn {"1D67} {\gamma}
151 \@@_setup_active_subscript:nn {"1D68} {\rho}
152 \@@_setup_active_subscript:nn {"1D69} {\phi}
153 \@@_setup_active_subscript:nn {"1D6A} {\chi}
154 </package>

```

File XIX

um-code-compat.dtx

20 Compatibility

```
1 (*package)
```

21 Patching/augmenting 3rd-party packages

21.1 url

Here we need to get url in a state such that when it switches to math mode and enters ASCII characters, the maths setup (i.e., unicode-math) doesn't remap the symbols into Plane 1. Which is what `\symliteral` is intended to do. This is the same as writing, e.g., `\def\UrlFont{\ttfamily\@@_switch_to:n{literal}}` but activates automatically so documents that might change the `\url` font through the standard interface still work correctly.

```
2 \@@_after_package:nNn {url} \@@_patch_url:
3 {
4   \tl_put_left:Nn \Url@FormatString { \@@_switch_to:n {literal} }
5   \tl_put_right:Nn \UrlSpecials
6   {
7     \do \` { \mathchar`` }
8     \do \' { \mathchar``' }
9     \do \$ { \mathchar``$ }
10    \do \& { \mathchar``& }
11  }
12 }
```

21.2 mathtools

mathtools's `\cramped` command and others that make use of its internal version use an incorrect font dimension.

The X_YTeX version is pretty similar to the legacy version, only using the correct font dimensions. Note we used `'\XeTeXradical'` with the family 255 to be almost sure that the radical rule width is not set. Former use of `'\newfam'` had an upsetting effect on legacy math alphabets.

```
13 (*XE)
14 \@@_after_package:nNn { mathtools } \@@_patch_mathtools_A:
15 {
16   \cs_set_nopar:Npn \MT_cramped_internal:Nn ##1 ##2
17   {
18     \hbox_set:Nn \l_tmpa_box
19     {
20       \color@setgroup \c_math_toggle_token \m@th
21       ##1
```

```

22         \dim_zero:N \nulldelimiterspace
23         \XeTeXradical 255 ~ 0 ~ { ##2 }
24         \c_math_toggle_token \color@endgroup
25     }
26     \box_set_ht:Nn \l_tmpa_box
27     {
28         \box_ht:N \l_tmpa_box - \@_radical_vgap:N ##1
29     }
30     \box_use_drop:N \l_tmpa_box
31 }
32 }
33 </XE>

```

`\overbracket` and `\underbracket` from `mathtools`'s `\overbracket` and `\underbracket` take optional arguments and are defined in terms of rules, so we keep them, and rename ours to `\Uoverbracket` and `\Underbracket`.

Original definition used the height of `\bracketd` which is not available with Unicode fonts, so we are hard coding the $5/18ex$ suggested by `mathtools`'s documentation.

```

34 \@_after_package:nNn { mathtools } \@_patch_mathtools_B:
35 {
36     \cs_set_eq:NN \MTOverbracket \overbracket
37     \cs_set_eq:NN \MTUnderbracket \underbracket
38
39     \AtBeginDocument
40     {
41         \msg_warning:nn { unicode-math } { mathtools-overbracket }
42
43         \cs_set:Npn \downbracketfill ##1 ##2
44         {
45             \tl_set:Nn \l_MT_bracketheight_fdim {.27ex}
46             \downbracketend {##1} {##2}
47             \leaders \vrule \@height ##1 \@depth \z@ \hfill
48             \downbracketend {##1} {##2}
49         }
50
51         \cs_set:Npn \upbracketfill ##1 ##2
52         {
53             \tl_set:Nn \l_MT_bracketheight_fdim {.27ex}
54             \upbracketend {##1} {##2}
55             \leaders \vrule \@height \z@ \@depth ##1 \hfill
56             \upbracketend {##1} {##2}
57         }
58
59         \cs_set_eq:NN \Uoverbracket \overbracket
60         \cs_set_eq:NN \Underbracket \underbracket
61         \cs_set_eq:NN \overbracket \MTOverbracket
62         \cs_set_eq:NN \underbracket \MTUnderbracket
63     }

```

```
64 }
```

`\dblcolon` `mathtools` defines several commands as combinations of colons and other characters, but with meanings incompatible to `unicode-math`. Thus we issue a warning.
`\coloneqq` Note `mathtools` uses `\providecommand` `\AtBeginDocument`.

```
\eqqcolon 65 \@@_after_package:nNn { mathtools } \@@_patch_mathtools_C:  
66 {  
67   \msg_warning:nn { unicode-math } { mathtools-colon }  
68   \DeclareDocumentCommand \dblcolon { } { \Colon }  
69   \DeclareDocumentCommand \coloneqq { } { \coloneq }  
70   \DeclareDocumentCommand \Coloneqq { } { \Coloneq }  
71   \DeclareDocumentCommand \eqqcolon { } { \eqcolon }  
72 }  
  
73 </package>
```

File XX

um-code-amsmath.dtx

22 *Compatibility with amsmath*

```
1 (*package)
```

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:

```
2 \tl_remove_once:Nn \@begindocumenthook
3 {
4   \mathchardef\std@minus\mathcode`\-\relax
5   \mathchardef\std@equal\mathcode`\=\relax
6 }
7 \AtBeginDocument
8 {
9   \Umathcharnumdef\std@minus\Umathcodenum`\-
10  \Umathcharnumdef\std@equal\Umathcodenum`\=
11 }
12 \cs_set:Npn \@cdots {\mathinner{\unicodecdots}}
13 \cs_set_eq:NN \dotsb@ \cdots
```

This isn't as clever as the `amsmath` definition but I think it works:

```
14 (*XE)
15 \def \resetMathstrut@
16 {%
17   \setbox\z@\hbox{${$}%}
18   \ht\Mathstrutbox@ \ht\z@ \dp\Mathstrutbox@ \dp\z@
19 }
```

The subarray environment uses inappropriate font dimensions.

```
20 \cs_set:Npn \subarray #1
21 {
22   \vcenter
23   \c_group_begin_token
24   \Let@
25   \restore@math@cr
26   \default@tag
27   \skip_set:Nn \baselineskip
28   {
29     \@@_stack_num_up:N \scriptstyle
30     + \@@_stack_denom_down:N \scriptstyle
31   }
32   \lineskip \@@_stack_vgap:N \scriptstyle
33   \lineskiplimit \lineskip
34   \ialign
35   \c_group_begin_token
36   \token_if_eq_meaning:NNT c #1 { \hfil }
37   \c_math_toggle_token
```



```

38     \m@th
39     \scriptstyle
40     \c_parameter_token \c_parameter_token
41     \c_math_toggle_token
42     \hfil
43     \c_rcr
44 }
45 </XE>

```

The roots need a complete rework.

```

46 <*LU>
47 \cs_set_nopar:Npn \plainroot@ #1 \of #2
48 {
49   \bool_if:nTF
50   {
51     \@@_int_if_zero_p:n \uproot@ && \@@_int_if_zero_p:n \leftroot@
52   }
53   {
54     \Uroot \c_@@_radical_sqrt_tl { #1 } { #2 }
55   }
56   {
57     \hbox_set:Nn \rootbox
58     {
59       \c_math_toggle_token \m@th
60       \scriptscriptstyle { #1 }
61       \c_math_toggle_token
62     }
63     \mathchoice
64     { \r@@@t \displaystyle { #2 } }
65     { \r@@@t \textstyle { #2 } }
66     { \r@@@t \scriptstyle { #2 } }
67     { \r@@@t \scriptscriptstyle { #2 } }
68   }
69   \c_group_end_token
70 }
71 </LU>
72 \cs_set_nopar:Npn \r@@@t #1 #2
73 <*LU>
74 {
75   \hbox_set:Nn \l_tmpa_box
76   {
77     \c_math_toggle_token \m@th
78     #1 \mskip \uproot@ mu
79     \c_math_toggle_token
80   }
81   \Uroot \c_@@_radical_sqrt_tl
82   {
83     \box_move_up:nn { \box_wd:N \l_tmpa_box }
84     {
85       \hbox:n

```

```

86         {
87         \c_math_toggle_token \m@th
88         \mkern -\leftroot@ mu
89         \box_use:N \rootbox
90         \mkern \leftroot@ mu
91         \c_math_toggle_token
92         }
93     }
94 }
95 { #2 }
96 }
97 </LU>
98 <*XE>
99 {
100 \hbox_set:Nn \l_tmpa_box
101 {
102 \c_math_toggle_token \m@th
103 #1 \sqrt{sign { #2 }}
104 \c_math_toggle_token
105 }
106 \hbox_set:Nn \l_tmpb_box
107 {
108 \c_math_toggle_token \m@th
109 #1 \mskip \uproot@ mu
110 \c_math_toggle_token
111 }
112 \mkern -\leftroot@ mu
113 \@_mathstyle_scale:NnnN #1 { \kern } { \fontdimen 63 \g_@@_sqrt_font_cmd_tl } \g_@@_sqrt_font_
114 \box_move_up:nn
115 {
116 \box_wd:N \l_tmpb_box + (\box_ht:N \l_tmpa_box - \box_dp:N \l_tmpa_box)
117 * \number \fontdimen 65 \g_@@_sqrt_font_cmd_tl / 100
118 }
119 { \box_use:N \rootbox }
120 \@_mathstyle_scale:NnnN #1 { \kern } { \fontdimen 64 \g_@@_sqrt_font_cmd_tl } \g_@@_sqrt_font_
121 \mkern \leftroot@ mu
122 \box_use_drop:N \l_tmpa_box
123 }
124 </XE>
125 </package>

```

File XXI

um-code-epilogue.dtx

23 Epilogue

1 (*package)

Lots of little things to tidy up.

23.1 Resolving Greek symbol name control sequences

\@@_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.

```
2 \AtBeginDocument { \debug_suspend: \@@_resolve_greek: \debug_resume: }
3 \cs_new:Npn \@@_resolve_greek:
4 {
5   \clist_map_inline:nn
6     {
7     Alpha,Beta,Gamma,Delta,Epsilon,Zeta,Eta,Theta,Iota,Kappa,Lambda,
8     alpha,beta,gamma,delta,epsilon,zeta,eta,theta,iota,kappa,lambda,
9     Mu,Nu,Xi,Omicron,Pi,Rho,Sigma,Tau,Upsilon,Phi,Chi,Psi,Omega,
10    mu,nu,xi,omicron,pi,rho,sigma,tau,upsilon,phi,chi,psi,omega,
11    varTheta,varsigma,vartheta,varkappa,varrho,varpi,varepsilon,varphi
12  }
13  {
14    \tl_set:cx {##1} { \exp_not:c { mit ##1 } }
15    \tl_set:cx {up ##1} { \exp_not:N \symup \exp_not:c { ##1 } }
16    \tl_set:cx {it ##1} { \exp_not:N \symit \exp_not:c { ##1 } }
17  }
18 }
```

23.1.1 Active fractions

Active fractions can be set up independently of any maths font definition; all it requires is a mapping from the Unicode input chars to the relevant L^AT_EX fraction declaration.

```
19 \cs_new:Nn \@@_which_frac:nn
20 {
21   \bool_if:NTF \l_@@_smallfrac_bool { \tfrac } { \frac } {#1} {#2}
22 }
23 \cs_new:Npn \@@_setup_active_frac:
24 {
25   \@@_mathactive_remap:nn {"2189} { \@@_which_frac:nn {0} {3} }
26   \@@_mathactive_remap:nn {"2152} { \@@_which_frac:nn {1} {10} }
27   \@@_mathactive_remap:nn {"2151} { \@@_which_frac:nn {1} {9} }
```

```

28 \@@_mathactive_remap:nn {"215B} { \@@_which_frac:nn {1} {8} }
29 \@@_mathactive_remap:nn {"2150} { \@@_which_frac:nn {1} {7} }
30 \@@_mathactive_remap:nn {"2159} { \@@_which_frac:nn {1} {6} }
31 \@@_mathactive_remap:nn {"2155} { \@@_which_frac:nn {1} {5} }
32 \@@_mathactive_remap:nn {"00BC} { \@@_which_frac:nn {1} {4} }
33 \@@_mathactive_remap:nn {"2153} { \@@_which_frac:nn {1} {3} }
34 \@@_mathactive_remap:nn {"215C} { \@@_which_frac:nn {3} {8} }
35 \@@_mathactive_remap:nn {"2156} { \@@_which_frac:nn {2} {5} }
36 \@@_mathactive_remap:nn {"00BD} { \@@_which_frac:nn {1} {2} }
37 \@@_mathactive_remap:nn {"2157} { \@@_which_frac:nn {3} {5} }
38 \@@_mathactive_remap:nn {"215D} { \@@_which_frac:nn {5} {8} }
39 \@@_mathactive_remap:nn {"2154} { \@@_which_frac:nn {2} {3} }
40 \@@_mathactive_remap:nn {"00BE} { \@@_which_frac:nn {3} {4} }
41 \@@_mathactive_remap:nn {"2158} { \@@_which_frac:nn {4} {5} }
42 \@@_mathactive_remap:nn {"215A} { \@@_which_frac:nn {5} {6} }
43 \@@_mathactive_remap:nn {"215E} { \@@_which_frac:nn {7} {8} }
44 }
45 \AtBeginDocument { \@@_setup_active_frac: }

```

23.2 *Synonyms and all the rest*

These are symbols with multiple names. Eventually to be taken care of automatically by the maths characters database.

```

46 \protected\def\to{\rightarrow}
47 \protected\def\le{\leq}
48 \protected\def\ge{\geq}
49 \protected\def\neq{\ne}
50 \protected\def\triangle{\mathord{\bigtriangleup}}
51 \protected\def\bigcirc{\mdlgwhtcircle}
52 \protected\def\circ{\vysmwhtcircle}
53 \protected\def\bullet{\smbkcircle}
54 \protected\def\mathyen{\yen}
55 \protected\def\mathsterling{\sterling}
56 \protected\def\diamond{\smwhtdiamond}
57 \protected\def\emptyset{\varnothing}
58 \protected\def\hbar{\hslash}
59 \protected\def\land{\wedge}
60 \protected\def\lor{\vee}
61 \protected\def\owns{\ni}
62 \protected\def\gets{\leftarrow}
63 \protected\def\mathring{\ocirc}
64 \protected\def\not{\neg}
65 \protected\def\longdivision{\longdivisionsign}

```

These are somewhat odd: (and their usual Unicode uprightness does not match their amssymb glyphs)

```

66 \protected\def\backepsilon{\upbackepsilon}
67 \protected\def\eth{\matheth}

```

These are names that are ‘frozen’ in HTML but have dumb names:

```
68 \protected\def\dbkarow {\dbkarrow}
69 \protected\def\drbkarow{\drbkarow}
70 \protected\def\hksearrow{\hksearrow}
71 \protected\def\hkswarrow{\hkswarrow}
```

Due to the magic of OpenType math, big operators are automatically enlarged when necessary. Since there isn’t a separate unicode glyph for ‘small integral’, I’m not sure if there is a better way to do this:

```
72 \protected\def\smallint{\mathop{\textstyle\int}\limits}
```

`\underbar`

```
73 \cs_set_eq:NN \latexe_underbar:n \underbar
74 \renewcommand\underbar
75 {
76   \mode_if_math:TF \mathunderbar \latexe_underbar:n
77 }
```

`\smallsetminus`

```
78 \cs_set_protected:Npn \smallsetminus { \mathbin{ \mathpalette \@@_smallset-
  minus: \relax} }
79 \cs_set:Npn \__um_smallsetminus: #1 { \rotatebox{135}{ \smash{ \raisebox{-
  \height}{\$#1\minus$} } }}}
```

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

```
\Digamma 80 \cs_set_protected:Npn \digamma {\updigamma}
81 \cs_set_protected:Npn \Digamma {\upDigamma}
```

Symbols

```
82 \cs_set_protected:Npn \lvert {\Vert}
  \mathinner items:
83 \cs_set_protected:Npn \mathellipsis {\mathinner{\unicodeellipsis}}
84 \cs_set_protected:Npn \cdots {\mathinner{\unicodcdots}}
85 \cs_set_eq:NN \@@_text_slash: \slash
86 \cs_set_protected:Npn \slash
87 {
88   \mode_if_math:TF {\mathslash} {\@@_text_slash:}
89 }
```

23.2.1 `\not`

The situation of `\not` symbol is currently messy, in Unicode it is defined as a combining mark so naturally it should be treated as a math accent, however X_{TeX} does not correctly place it as it needs special treatment compared to other accents. Furthermore a math accent changes the spacing of its nucleus, so `\not=` will be spaced as an ordinary not relational symbol, which is undesired.

Here modify `\not` to a macro that tries to use predefined negated symbols, which would give better results in most cases, until there is more robust solution in the engines.

This code is based on an answer to a TeX – Stack Exchange question by Enrico Gregorio³.

`\not`

```

90 \DeclareDocumentCommand \not {m}
91 {
92   \tl_set:Nx \l_@@_not_token_name_tl { \cs_to_str:N #1 }
93   \tl_if_empty:NT \l_@@_not_token_name_tl
94   {
95     \tl_set:Nx \l_@@_not_token_name_tl { \token_to_str:N #1 }
96   }
97   \cs_if_exist:cTF { not \l_@@_not_token_name_tl }
98   {
99     \use:c { not \l_@@_not_token_name_tl }
100  }
101  {
102    \cs_if_exist:cTF { n \l_@@_not_token_name_tl }
103    {
104      \use:c { n \l_@@_not_token_name_tl }
105    }
106    {
107      \tl_if_eq:nnTF {#1} {$} { \notaccent{} } { \notaccent } #1
108    }
109  }
110 }

```

`\NewNegationCommand`

`\RenewNegationCommand`

```

111 \DeclareDocumentCommand \NewNegationCommand {mm}
112 {
113   \@@_set_negation_command:Nnn \cs_new_protected:cpn {#1} {#2}
114 }
115 \DeclareDocumentCommand \RenewNegationCommand {mm}
116 {
117   \@@_set_negation_command:Nnn \cs_set_protected:cpn {#1} {#2}
118 }
119 \cs_set:Nn \@@_set_negation_command:Nnn
120 {
121   \tl_set:Nx \l_@@_not_token_name_tl { \cs_to_str:N #2 }
122   \tl_if_empty:NT \l_@@_not_token_name_tl
123   {
124     \tl_set:Nx \l_@@_not_token_name_tl { \token_to_str:N #2 }
125   }
126   #1 { not \l_@@_not_token_name_tl } { #3 }
127 }

```

³<http://tex.stackexchange.com/a/47260/729>

```

128 \NewNegationCommand { = } { \neq }
129 \NewNegationCommand { < } { \less }
130 \NewNegationCommand { > } { \ngtr }
131 \NewNegationCommand { \gets } { \leftarrow }
132 \NewNegationCommand { \simeq } { \nsime }
133 \NewNegationCommand { \equal } { \ne }
134 \NewNegationCommand { \le } { \leq }
135 \NewNegationCommand { \ge } { \ngeq }
136 \NewNegationCommand { \greater } { \ngtr }
137 \NewNegationCommand { \forksnot } { \forks }

```

23.2.2 Full-width remapping

While this could be done with the full mathcode remapping machinery used for the other purposes, it would be fairly redundant with plain ASCII. Worse, this would slow down what is already an inefficient part of unicode-math.

Instead we use mathactive to do a plain old mapping from full-width to ASCII directly.

Until I get requests for it, I've not included symbols or punctuation here.

Numbers

```

138 \int_step_inline:nnnn {0} {1} {9}
139 {
140   \@@_mathactive_remap:nn {"FF10+#1} {\char\int_eval:n{\0+#1}}
141 }

```

Letters

```

142 \int_step_inline:nnnn {0} {1} {26}
143 {
144   \@@_mathactive_remap:nn {"FF21+#1} {\char\int_eval:n{\A+#1}}
145   \@@_mathactive_remap:nn {"FF41+#1} {\char\int_eval:n{\a+#1}}
146 }

```

23.3 Legacy characters

\@@_undeclare_symbol:N

```

147 \cs_new:Nn \@@_undeclare_symbol:N
148 {
149   \cs_set_protected:Npn #1
150     { \@@_error:nx {legacy-char-not-supported} { \token_to_str:N #1 } }
151 }

```

If you have better ideas about what to do here, please mention.

```

152 \@@_undeclare_symbol:N \arrowvert
153 \@@_undeclare_symbol:N \Arrowvert
154 \@@_undeclare_symbol:N \bracevert
155 </package>

```

Fin

The official end of the package:

156 `<package>\endinput`

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	
\backslash \$	9
\backslash &	10
\backslash '	8, 260, 277
\backslash *	252
\backslash -	4, 251
\backslash .	236
\backslash /	304
\backslash :	255
\backslash ::	16
\backslash ::N	16
\backslash ::x_unbraced	16
\backslash <	308
\backslash =	5
\backslash >	309
\backslash @@@math@bgroup	53
\backslash @@@math@egroup	54
\backslash @_accent:nnn	65, 131, 138
\backslash @_after_package:nNn	2, 8, 14, 34, 65
\backslash @_alphabet_config:nnn	123, 129, 141, 169
\backslash @_arg_i_before_egroup:n	2, 6
\backslash @_assign_delcode:n	301, 307, 310–336
\backslash @_assign_delcode:nn	45, 69, 290, 301, 304–306, 308, 309
\backslash @_assign_delcode_noparse:nn	69, 290, 298
\backslash @_assign_delcode_parse:nn	45, 294
\backslash @_backprime_double_mchar	48, 256, 275
\backslash @_backprime_single_mchar	168, 174, 180, 185, 190, 195, 255, 274
\backslash @_backprime_triple_mchar	53, 257, 276
\backslash @_char_gmake_mathactive:N	69
\backslash @_char_gmake_mathactive:n	69, 75, 81, 288
\backslash @_check_math_alphabet:	79, 83
\backslash @_copy_fontdimen:nnN	89, 178–190, 215–219
\backslash @_cs_new:Nn	2, 2, 2, 6, 10, 10, 12, 19, 23, 24, 29, 30, 32, 38, 39, 44, 48, 73, 75, 85, 110, 110, 136, 153, 168, 206, 234, 249, 268, 302, 348, 353
\backslash @_cs_set:Nn	60, 69, 78, 95, 101, 107, 112, 152
\backslash @_debug:n	116, 138, 145
\backslash @_debug_end:n	147, 159, 168
\backslash @_debug_start:n	142, 157, 166
\backslash @_declare_math_sizes:	13, 72
\backslash @_default_mathalph:nnn	29, 35–51
\backslash @_define_math_chars:	11, 12
\backslash @_define_prime_chars:	248, 269
\backslash @_define_prime_commands:	248, 249
\backslash @_delimiter:Nnn	61, 109, 117, 124, 126
\backslash @_error:n	21, 78, 127
\backslash @_error:nx	112, 128, 150
\backslash @_font_dimen:Nnnnn	12, 78
\backslash @_font_param:n	70, 103, 110, 113, 122, 135, 145
\backslash @_font_param:nn	66, 101, 102, 108, 109, 111, 112, 114–121, 123, 124, 126–129, 133, 134, 136–139, 141–144
\backslash @_font_param:nnn	62, 104–107, 125, 130–132
\backslash @_font_param:nnnnn	47, 64, 68, 72, 140
\backslash @_font_param_aux:NNN	73
\backslash @_font_param_aux:NNnnnn	73
\backslash @_font_param_aux:ccc	58
\backslash @_font_param_aux:ccnnnn	50
\backslash @_fontdimen_pc_to_pt:nN	73, 74, 146
\backslash @_fontdimen_to_percent:nN	146, 157, 160
\backslash @_fontface_gset_eq:NN	32, 100, 146
\backslash @_fontface_gset_eq:cN	142
\backslash @_fontspec_select_font:	15, 110
\backslash @_fontspec_trial_font:	12, 85
\backslash @_fontswitch:n	43, 45
\backslash @_glyph_if_exist:Nn	24
\backslash @_glyph_if_exist:NnT	96
\backslash @_glyph_if_exist:NnTF	22, 24, 27, 32, 47, 52, 126
\backslash @_group_begin:	8, 20, 22, 48, 51
\backslash @_group_begin_frozen:	22, 51
\backslash @_group_end:n	16, 21, 23, 49, 52, 56
\backslash @_group_end_frozen:n	23, 52
\backslash @_if_alphabet_exists:nn	173

$\backslash@@_if_alphabet_exists:nnT$ 118
 $\backslash@@_if_alphabet_exists:nnTF$ 88
 $\backslash@@_if_char_spec:nNT$
. [119](#), [184](#), [244](#), [260](#), [282](#), [296](#)
 $\backslash@@_init:n$ [5](#), [48](#)
 $\backslash@@_init_alphabet:n$ [5](#), [19](#), [68](#), [74](#)
 $\backslash@@_input_math_symbol_table:$
. [10](#), [23](#), [24](#), [33](#)
 $\backslash@@_int_if_slot_in_range:nnT$
. [48](#), [131](#), [147](#)
 $\backslash@@_int_if_slot_is_last_in_range:nnT$
. [128](#), [162](#)
 $\backslash@@_int_if_zero:n$ [97](#)
 $\backslash@@_int_if_zero:nTF$ [97](#)
 $\backslash@@_int_if_zero_p:n$ [51](#), [97](#)
 $\backslash@@_keys_choices:nn$
. [2](#), [26](#), [31](#), [36](#), [41](#), [46](#), [51](#), [56](#), [90](#),
[113](#), [140](#), [146](#), [160](#), [174](#), [179](#), [185](#), [207](#)
 $\backslash@@_keys_choices_aux:nnn$ [4](#), [14](#)
 $\backslash@@_keys_choices_fn:nn$ [4](#), [10](#)
 $\backslash@@_load_lm:$ [43](#), [44](#)
 $\backslash@@_log:n$ [12](#), [21](#), [25](#), [30](#), [37](#), [131](#)
 $\backslash@@_log:nx$ [21](#), [122](#), [128](#), [132](#)
 $\backslash@@_make_mathactive:nNN$
. [46](#), [70](#), [270](#)–[278](#), [280](#)
 $\backslash@@_make_mathactive_noparse:nNN$
. [70](#), [283](#), [285](#)
 $\backslash@@_make_mathactive_parse:nNN$ [46](#), [280](#)
 $\backslash@@_map_char_noparse:nn$. . . [19](#), [28](#), [178](#)
 $\backslash@@_map_char_parse:nn$ [60](#), [178](#)
 $\backslash@@_map_char_single:nn$
. [19](#), [28](#), [60](#), [178](#), [189](#), [195](#)
 $\backslash@@_map_char_single:nnn$
. [187](#), [212](#), [241](#)–[246](#), [261](#)
 $\backslash@@_map_chars_range:nnn$ [192](#), [202](#)
 $\backslash@@_map_chars_range:nnnn$
. [192](#), [224](#), [232](#), [240](#), [260](#), [268](#)
 $\backslash@@_mathactive_remap:nn$
. [5](#), [16](#), [25](#)–[43](#), [77](#), [140](#), [144](#), [145](#)
 $\backslash@@_mathalph_decl:nF$ [55](#), [58](#)
 $\backslash@@_mathalph_map:$ [20](#), [29](#), [61](#), [63](#)
 $\backslash@@_mathgroup_set:n$ [15](#), [37](#), [85](#)
 $\backslash@@_mathmap_noparse:nnn$ [18](#), [27](#), [270](#), [281](#)
 $\backslash@@_mathmap_parse:nnn$ [59](#), [277](#)
 $\backslash@@_mathparam_restore:$ [40](#), [348](#)
 $\backslash@@_mathparam_store:$ [36](#), [353](#)
 $\backslash@@_mathparam_store_aux:N$ [358](#)–[361](#), [364](#)
 $\backslash@@_mathstyle_scale:NnnN$. [113](#), [120](#), [154](#)

$\backslash@@_maybe_init_alphabet:n$
. [15](#)–[17](#), [44](#), [68](#), [74](#), [92](#), [99](#)
 $\backslash@@_msg_new:nn$ [3](#),
[7](#), [11](#), [15](#), [19](#), [23](#), [28](#), [32](#), [38](#), [42](#),
[46](#), [50](#), [59](#), [69](#), [84](#), [89](#), [93](#), [99](#), [104](#), [135](#)
 $\backslash@@_nbackprimes_select:nn$ [41](#), [239](#)
 $\backslash@@_new_alphabet_config:nnn$
. [2](#), [8](#), [19](#), [36](#), [47](#), [61](#), [88](#), [97](#), [118](#),
[130](#), [142](#), [148](#), [167](#), [172](#), [184](#), [189](#),
[198](#), [206](#), [219](#), [226](#), [238](#), [245](#), [254](#),
[258](#), [263](#), [280](#), [296](#), [313](#), [329](#), [333](#),
[337](#), [341](#), [362](#), [383](#), [400](#), [417](#), [458](#),
[464](#), [485](#), [506](#), [523](#), [540](#), [589](#), [594](#),
[598](#), [602](#), [606](#), [610](#), [614](#), [619](#), [636](#),
[653](#), [670](#), [687](#), [732](#), [749](#), [766](#), [783](#), [800](#)
 $\backslash@@_new_cramped_style:N$ [2](#), [8](#)–[11](#)
 $\backslash@@_nprimes:Nn$ [8](#), [24](#), [29](#), [34](#), [38](#), [49](#), [54](#), [58](#)
 $\backslash@@_nprimes_select:nn$ [16](#), [153](#)
 $\backslash@@_numrange_last_parse:nwT$. [164](#), [166](#)
 $\backslash@@_numrange_parse:nwT$ [149](#), [151](#)
 $\backslash@@_onceoff_setup:$ [25](#), [234](#)
 $\backslash@@_patch_mathtools_A:$ [14](#)
 $\backslash@@_patch_mathtools_B:$ [34](#)
 $\backslash@@_patch_mathtools_C:$ [65](#)
 $\backslash@@_patch_url:$ [2](#)
 $\backslash@@_prepare_mathstyle:n$ [2](#), [59](#)
 $\backslash@@_prime_double_mchar$. . . [23](#), [252](#), [271](#)
 $\backslash@@_prime_quad_mchar$ [33](#), [254](#), [273](#)
 $\backslash@@_prime_single_mchar$ [65](#),
[71](#), [77](#), [83](#), [88](#), [93](#), [98](#), [103](#), [251](#), [270](#)
 $\backslash@@_prime_triple_mchar$. . . [28](#), [253](#), [272](#)
 $\backslash@@_print_indent:n$ [35](#), [37](#)
 $\backslash@@_process_symbol_noparse:nnn$ [66](#), [238](#)
 $\backslash@@_process_symbol_parse:nnn$. [42](#), [238](#)
 $\backslash@@_radical:nn$ [57](#), [98](#)
 $\backslash@@_radical_vgap:N$ [28](#)
 $\backslash@@_range_decl:n$ [55](#), [101](#)
 $\backslash@@_range_init:$ [23](#), [38](#)
 $\backslash@@_range_process:$ [51](#)
 $\backslash@@_range_process:n$ [34](#), [51](#)
 $\backslash@@_remap_symbol:nnn$ [43](#), [67](#), [251](#), [252](#), [255](#)
 $\backslash@@_remap_symbol_noparse:nnn$. [67](#), [258](#)
 $\backslash@@_remap_symbol_parse:nnn$. . . [43](#), [258](#)
 $\backslash@@_remap_symbols:$ [26](#), [249](#)
 $\backslash@@_resolve_greek:$ [2](#)
 $\backslash@@_rm_series_default:n$
. [87](#), [97](#)–[99](#), [103](#), [104](#), [109](#), [110](#), [114](#), [115](#)
 $\backslash@@_scan_backdprime:$ [170](#), [215](#)
 $\backslash@@_scan_backprime:$ [164](#), [205](#)

<code>\@@_scan_backtrprime:</code>	176, 227	139, 239, 339, 402, 406, 412, 525,
<code>\@@_scan_dprime:</code>	67, 117	529, 535, 675, 681, 684, 788, 794, 797
<code>\@@_scan_prime:</code>	61, 111	<code>\@@_set_mathalphabet_latin:nnn</code> . .
<code>\@@_scan_qprime:</code>	79, 141 31–33,
<code>\@@_scan_sscript:</code>	10, 21, 24, 27	113, 169, 221, 231, 240, 256, 285,
<code>\@@_scan_sscript:TF</code>	26, 30	291, 294, 318, 323, 324, 327, 339,
<code>\@@_scan_sscript_aux:nnN</code>	35, 39	368, 372, 378, 491, 495, 501, 596,
<code>\@@_scan_sup_backdprime:</code>	187, 278	604, 612, 641, 647, 650, 754, 760, 763
<code>\@@_scan_sup_backprime:</code>	182, 276, 277	<code>\@@_set_mathalphabet_numbers:nnn</code> .
<code>\@@_scan_sup_backtrprime:</code>	192, 279 5, 186, 260,
<code>\@@_scan_sup_dprime:</code>	90, 273	261, 267, 313, 331, 460, 461, 616, 617
<code>\@@_scan_sup_prime:</code>	85, 271, 272, 283	<code>\@@_set_mathalphabet_pos:nnnn</code> . . .
<code>\@@_scan_sup_qprime:</code>	100, 275 83–86, 114,
<code>\@@_scan_sup_trprime:</code>	95, 274	115, 164, 165, 175–181, 191–195,
<code>\@@_scan_trprime:</code>	73, 129	200–204, 209, 210, 211, 211–216,
<code>\@@_scanbackprime_collect:N</code>	222–224, 229–236, 241–243, 247,
.	168, 174, 180, 185, 190, 195,	248, 248, 249, 249, 250, 250, 251,
.	197, 202, 207, 212, 218, 224, 230, 236	251, 252, 252, 262, 302, 433, 434,
<code>\@@_scanprime_collect:N</code>	65,	438, 443, 449, 454, 562–567, 570,
.	71, 77, 83, 88, 93, 98, 103, 105, 109,	575, 580, 585, 709, 710, 713, 718,
.	112, 115, 120, 126, 132, 138, 144, 150	723, 728, 822, 823, 826, 831, 836, 841
<code>\@@_set_big_operator:nnn</code>	45, 73	<code>\@@_set_mathchar:NNnn</code> 47, 52, 287
<code>\@@_set_delcode:nnn</code>	<code>\@@_set_mathchar:cNnn</code> 47, 79
.	53, 106, 114, 122, 236, 292	<code>\@@_set_mathcode:nnn</code>
<code>\@@_set_math_accent:Nnnn</code> 37, 40–44, 107, 115, 121
.	50, 52, 54, 56, 62, 63, 128	<code>\@@_set_mathcode:nnnn</code> 37, 180, 266, 274
<code>\@@_set_math_close:nnn</code>	47, 112	<code>\@@_set_mathsymbol:nNnn</code> 27, 240
<code>\@@_set_math_fence:nnnn</code>	48, 119	<code>\@@_set_negation_command:Nnn</code>
<code>\@@_set_math_open:nnn</code>	46, 87 113, 117, 119
<code>\@@_set_math_overunder:Nnnn</code> 58, 60, 133		<code>\@@_set_normal_Greek:nn</code>
<code>\@@_set_mathalph_range:nnnn</code>	289, 294 38, 40, 122, 125, 255,
<code>\@@_set_mathalph_range:nnnnn</code>	388, 394, 511, 517, 657, 663, 770, 776
.	294, 316, 321, 327, 335, 343	<code>\@@_set_normal_Latin:nn</code>
<code>\@@_set_mathalphabet_Greek:nnn</code> 10, 12, 90, 92,
.	42–44,	219, 267, 273, 300, 306, 345, 350,
.	127, 259, 331, 385, 389, 395, 508,	356, 468, 473, 479, 623, 629, 736, 742
.	512, 518, 658, 664, 667, 771, 777, 780	<code>\@@_set_normal_char:nnn</code>
<code>\@@_set_mathalphabet_Latin:nnn</code> 26–28, 65, 70, 75, 80, 102, 108–
.	14–16,	110, 146, 151, 156, 161, 205, 420,
.	94, 174, 208, 223, 228, 247, 268,	423, 427, 430, 544, 549, 554, 559,
.	274, 277, 301, 307, 310, 318, 335,	691, 696, 701, 706, 804, 809, 814, 819
.	347, 351, 357, 470, 474, 480, 591,	<code>\@@_set_normal_greek:nn</code>
.	600, 608, 624, 630, 633, 737, 743, 746 49, 53, 134, 137, 235,
<code>\@@_set_mathalphabet_char:nnn</code>	405, 411, 528, 534, 674, 680, 787, 793
.	18, 27, 59, 270, 286, 292	<code>\@@_set_normal_latin:nn</code>
<code>\@@_set_mathalphabet_char:nnnn</code> 21, 25, 101, 107,
.	284, 307, 328, 336, 344–349	227, 284, 290, 317, 323, 366, 371,
<code>\@@_set_mathalphabet_greek:nnn</code>	377, 489, 494, 500, 640, 646, 753, 759
.	56–58,	<code>\@@_set_normal_numbers:nn</code> 4, 265

<code>\clist_map_inline:nn</code>	5, 52, 53, 61, 64, 75, 203, 209, 221, 229, 237, 257, 265, 306, 315, 320, 325, 333, 341	<code>\cs_new:Npn</code>	3, 23, 61, 67, 73, 79, 85, 90, 95, 100, 127–132, 152, 162, 164, 170, 176, 182, 187, 192
<code>\clist_new:N</code>	75–82	<code>\cs_new_eq:Nc</code>	6
<code>\clist_put_right:Nx</code>	142	<code>\cs_new_nopar:Npn</code>	13, 76, 90
<code>\clist_set:No</code>	38, 68	<code>\cs_new_protected:cpn</code>	113
<code>\Colon</code>	68	<code>\cs_new_protected:Nn</code>	25, 85, 101, 164
<code>\coloncolon</code>	73	<code>\cs_new_protected:Npn</code>	12
<code>\coloncolonequals</code>	77	<code>\cs_new_protected_nopar:Nn</code>	
<code>\Coloneq</code>	70		2, 47, 62, 66, 70, 74, 88
<code>\coloneq</code>	69	<code>\cs_new_protected_nopar:Npn</code>	80, 94
<code>\Coloneqq</code>	64, 65	<code>\cs_set:cpx</code>	70
<code>\coloneqq</code>	63, 65	<code>\cs_set:Nn</code>	27, 45, 63, 119, 146, 192, 205, 238, 242, 364
<code>\colonequals</code>	75	<code>\cs_set:Npn</code>	
<code>\colonsep</code>	80		4, 12, 15, 20, 27, 29, 31, 32, 37, 42, 43, 47, 49, 51, 79, 91, 96, 151, 166
<code>\color@endgroup</code>	24	<code>\cs_set_eq:cc</code>	63
<code>\color@setgroup</code>	20	<code>\cs_set_eq:NN</code>	6, 8, 13, 18, 19, 19, 20, 22, 23, 27, 28, 36, 37, 42–46, 47, 48, 51–54, 56, 59, 59, 60, 60, 61, 62, 63, 66–68, 69, 69, 70, 73, 74, 75, 81, 85, 166, 172, 178, 251–257, 271–279, 283
<code>\crampeddisplaystyle</code>	8, 18	<code>\cs_set_nopar:Npn</code>	16, 47, 72
<code>\crampedscriptscriptstyle</code>	8	<code>\cs_set_protected:cpn</code>	23, 117
<code>\crampedscriptstyle</code>	8, 30	<code>\cs_set_protected:cpx</code>	6, 78
<code>\crampedtextstyle</code>	8, 24	<code>\cs_set_protected:Nn</code>	32, 155
<code>\rcrcr</code>	43	<code>\cs_set_protected:Npn</code>	8, 21, 40, 78, 80, 80, 81–84, 86, 86, 87, 149
<code>\cs_generate_variant:Nn</code>		<code>\cs_set_protected_nopar:Npn</code>	16
	17–19, 36, 52, 59, 85, 99, 133, 134	<code>\cs_set_protected_nopar:Npx</code>	76
<code>\cs_gset:cpx</code>	81	<code>\cs_to_str:N</code>	
<code>\cs_gset:Nn</code>	269		6, 76, 79, 81, 83, 92, 94, 96, 100, 102, 121, 123, 125, 157, 159, 166, 168
<code>\cs_gset_eq:NN</code>	21, 27–29	<code>\csname</code>	40, 45, 50, 55, 59, 63, 90, 92
<code>\cs_gset_protected_nopar:cpx</code>		<code>\cuberoot</code>	346
	96, 123, 125		
<code>\cs_gset_protected_nopar:Npx</code>			
	94, 108, 116, 130, 135		
<code>\cs_if_eq:cNF</code>	35		
<code>\cs_if_eq:NNT</code>	140		
<code>\cs_if_eq:NNTF</code>	15, 18, 21, 24, 27, 30, 33		
<code>\cs_if_exist:cF</code>	7, 155		
<code>\cs_if_exist:cTF</code>	97, 102, 105, 175		
<code>\cs_if_exist:NF</code>	5, 91, 154		
<code>\cs_if_exist:NT</code>	217		
<code>\cs_new:cn</code>	160		
<code>\cs_new:Nn</code>	2, 2, 2, 3, 8, 13, 14, 15, 16, 19, 19, 37, 41, 51, 53, 57, 58, 61, 65, 69, 73, 73, 73, 74, 77, 83, 87, 87, 89, 93, 101, 102, 105, 112, 119, 119, 128, 133, 135, 138, 142, 147, 147, 147, 148, 150, 154, 162, 169, 178, 182, 187, 197, 200, 219, 227, 235, 249, 255, 258, 263, 265, 270, 277, 280, 284, 285, 289, 290, 294, 294, 301, 302, 313, 318, 323, 331, 339		

D	
<code>\D</code>	142
<code>\d</code>	168
<code>\date</code>	46
<code>\dbkarow</code>	68
<code>\dbkarrow</code>	68
<code>\dblcolon</code>	62, 65
<code>\debug_resume:</code>	2
<code>\debug_suspend:</code>	2
<code>\DeclareDocumentCommand</code>	
	68–71, 90, 111, 115
<code>\DeclareMathAccent</code>	118

G

<code>\g</code>	170	<code>\g_@@_literal_Nabla_bool</code>	20, 63, 144, 149, 153, 157,
<code>\g_@@_alphabet_clist</code>	77		419, 447, 542, 578, 689, 721, 802, 834
<code>\g_@@_bad_alpha_clist</code>	78, 82–85	<code>\g_@@_literal_partial_bool</code>	21, 73, 154, 163, 167, 171,
<code>\g_@@_bfit_Greek_usv</code>	80		426, 436, 552, 568, 699, 711, 812, 824
<code>\g_@@_bfit_greek_usv</code>	81	<code>\g_@@_main_font_cmd_tl</code>	72, 150, 178–190, 215–219
<code>\g_@@_bfit_Latin_usv</code>	78	<code>\g_@@_main_font_defined_bool</code>	3, 21, 35, 43
<code>\g_@@_bfit_latin_usv</code>	79	<code>\g_@@_mathalph_seq</code>	6, 13, 34, 49, 50, 63, 65, 82, 87
<code>\g_@@_bfliteral_bool</code>	11, 116, 123, 130, 137,	<code>\g_@@_mathbf_text_bool</code>	26, 43, 44
	348, 369, 386, 403, 471, 492, 509, 526	<code>\g_@@_mathclasses_seq</code>	25, 90
<code>\g_@@_bfsfit_Greek_usv</code>	76	<code>\g_@@_mathit_text_bool</code>	25, 38, 39
<code>\g_@@_bfsfit_greek_usv</code>	77	<code>\g_@@_mathparam_settings_tl</code> 58, 351, 356	
<code>\g_@@_bfsfit_Latin_usv</code>	74	<code>\g_@@_mathparam_store_tl</code>	72
<code>\g_@@_bfsfit_latin_usv</code>	75	<code>\g_@@_mathrm_text_bool</code> 24, 28, 29, 33, 34	
<code>\g_@@_bfsfup_Greek_usv</code>	76	<code>\g_@@_mathsf_text_bool</code>	27, 48, 49
<code>\g_@@_bfsfup_greek_usv</code>	77	<code>\g_@@_mathstyles_seq</code>	4, 100
<code>\g_@@_bfsfup_Latin_usv</code>	74	<code>\g_@@_mathtable_tl</code>	8, 10, 60
<code>\g_@@_bfsfup_latin_usv</code>	75	<code>\g_@@_mathtt_text_bool</code>	28, 53, 54
<code>\g_@@_bfup_Greek_usv</code>	80	<code>\g_@@_mclass_range_seq</code> 48, 89, 109, 122	
<code>\g_@@_bfup_greek_usv</code>	81	<code>\g_@@_mversion_tl</code>	62
<code>\g_@@_bfup_Latin_usv</code>	78	<code>\g_@@_operator_mathfont_tl</code> .	14, 43, 56
<code>\g_@@_bfup_latin_usv</code>	79	<code>\g_@@_ot_math_bool</code>	2, 51, 104
<code>\g_@@_bfupGreek_bool</code>	14, 80, 117, 124, 131, 392, 515	<code>\g_@@_prime_font_cmd_tl</code>	22, 27, 32, 47, 52, 71, 152
<code>\g_@@_bfupgreek_bool</code>	15, 81, 118, 125, 132, 409, 532	<code>\g_@@_primekern_muskip</code> ...	13, 103, 104
<code>\g_@@_bfupLatin_bool</code>	12,	<code>\g_@@_remap_style_tl</code>	67
	78, 119, 126, 133, 343, 354, 466, 477	<code>\g_@@_sfliteral_bool</code>	17, 144, 265, 282, 298, 315,
<code>\g_@@_bfuplatin_bool</code>	13,		621, 638, 655, 672, 734, 751, 768, 785
	79, 120, 127, 134, 364, 375, 487, 498	<code>\g_@@_size_sfssf_fp</code> ..	78, 106, 123, 128
<code>\g_@@_char_nrange_clist</code>	75	<code>\g_@@_size_tfsf_fp</code> ...	77, 105, 120, 123
<code>\g_@@_char_range_seq</code>	47, 61, 88, 110, 116, 126, 129	<code>\g_@@_slash_delimiter_usv</code>	57, 181–183, 304–306
<code>\g_@@_curr_font_cmd_tl</code>	34, 69, 96, 126, 143	<code>\g_@@_sqrt_font_cmd_tl</code>	70, 113, 117, 120, 151
<code>\g_@@_debug_nest_int</code> .	137, 140, 144, 149	<code>\g_@@_style_tl</code>	66
<code>\g_@@_default_mathalph_seq</code>	13, 32, 35, 99	<code>\g_@@_subs_prop</code>	15, 102
<code>\g_@@_fam_int</code>	29, 40, 41	<code>\g_@@_supers_prop</code>	4, 101
<code>\g_@@_family_tl</code>	65	<code>\g_@@_symfont_tl</code>	63
<code>\g_@@_fontname_tl</code>	61	<code>\g_@@_trial_font</code>	73, 74, 80, 100
<code>\g_@@_fonts_used_int</code> .	30, 139, 142, 143	<code>\g_@@_unknown_keys_clist</code>	76
<code>\g_@@_init_bool</code>	4,	<code>\g_@@_upGreek_bool</code>	9, 40, 60, 67, 74, 76, 81, 125
	4, 12, 13, 17, 22, 25, 33, 60, 105, 144	<code>\g_@@_upgreek_bool</code>	10, 51, 61, 68, 75, 77, 82, 137
<code>\g_@@_literal_bool</code> ...	6, 10, 21, 38,		
	49, 59, 66, 73, 80, 87, 90, 99, 120, 132		
<code>\g_@@_literal_colon_bool</code>	23, 176, 177, 253		

K	
<code>\kern</code>	113, 120
<code>\keys_define:nn</code>	2, 2, 7, 10, 17, 199
<code>\keys_set:nn</code>	2
<code>\keys_set_known:nnN</code>	10, 10
L	
<code>\L</code>	147
<code>\l_@@_alphabet_clist</code>	38, 40, 68, 72, 76, 81, 85, 112
<code>\l_@@_alphabet_tl</code>	40, 87, 88, 90, 97, 106, 114, 116, 118, 120, 122, 123, 126, 128, 129, 137, 141
<code>\l_@@_curr_named_slot</code>	46, 163, 197, 214, 296, 309
<code>\l_@@_fam_three_tl</code> ...	45, 208, 226, 231
<code>\l_@@_fam_two_tl</code>	44, 170, 198, 203
<code>\l_@@_family_tl</code> 39, 137, 141, 158, 161, 165	
<code>\l_@@_font</code>	146, 150–152
<code>\l_@@_font_keyval_tl</code>	64, 87, 96, 112, 137, 172, 210
<code>\l_@@_fontname_tl</code>	5, 41, 50, 57, 58, 96, 137, 195, 223
<code>\l_@@_implicit_alph_bool</code> .	5, 14, 26, 132
<code>\l_@@_keyval_clist</code>	10, 13, 80
<code>\l_@@_mathmap_charints_clist</code>	7, 46, 62, 78, 142, 279
<code>\l_@@_mathstyle_tl</code>	4, 12, 50
<code>\l_@@_mathtable_tl</code>	3, 8, 59
<code>\l_@@_missing_alph_seq</code>	21, 30, 35, 64, 86, 134
<code>\l_@@_mversion_tl</code>	4, 6, 7, 8, 16, 18, 22, 52, 160, 162, 197, 200, 225, 228
<code>\l_@@_nolimits_tl</code>	19, 23, 52, 84, 338
<code>\l_@@_not_token_name_tl</code> 68, 92, 93, 95, 97, 99, 102, 104, 121, 122, 124, 126	
<code>\l_@@_primecount_int</code>	31, 64, 70, 76, 82, 87, 92, 97, 102, 107, 119, 125, 131, 137, 143, 149, 153, 167, 173, 179, 184, 189, 194, 199, 217, 223, 229, 235, 239
<code>\l_@@_radicals_tl</code>	51, 89, 346
<code>\l_@@_remap_style_tl</code> 43, 69, 123, 126, 129	
<code>\l_@@_script_features_tl</code> ...	12, 55, 125
<code>\l_@@_script_font_tl</code>	14, 57, 124
<code>\l_@@_smallfrac_bool</code> ..	21, 22, 189, 195
<code>\l_@@_ss_chain_tl</code>	7, 18, 28, 46, 54
<code>\l_@@_sscript_features_tl</code> ..	13, 56, 130
<code>\l_@@_sscript_font_tl</code>	15, 58, 129
<code>\l_@@_style_tl</code>	37, 38, 42, 44, 67, 75, 88, 92, 97, 99, 106, 116, 118, 122, 123, 128, 129, 136, 141
<code>\l_@@_symfont_label_tl</code>	41, 42, 53, 155, 157, 160, 164, 180, 240, 266, 274, 287, 292
<code>\l_@@_tmpa_key_tl</code>	41, 43, 55
<code>\l_@@_tmpa_tl</code>	9, 12, 19, 20, 22, 27–29, 42, 47, 56, 57, 59, 60, 64, 65, 67, 68, 70–73, 76, 78, 80, 84, 93, 98
<code>\l_@@_tmpb_tl</code>	44, 46, 48, 61, 85, 99
<code>\l_@@_tmpc_tl</code>	49, 62, 75, 76, 86, 94
<code>\l_@@_trial_family_tl</code>	53, 96, 99
<code>\l_@@_unknown_keys_clist</code> 10, 79, 93, 134	
<code>\l_MT_bracketheight_fdim</code>	45, 53
<code>\l_tmpa_box</code>	18, 26, 28, 30, 75, 83, 100, 116, 122
<code>\l_tmpb_box</code>	106, 116
<code>\land</code>	59
<code>\latex_underbar:n</code>	73, 76
<code>\le</code>	47, 134
<code>\leaders</code>	47, 55
<code>\leftarrow</code>	62
<code>\leftroot@</code>	51, 88, 90, 112, 121
<code>\leq</code>	47
<code>\Let@</code>	24
<code>\limits</code>	72, 140
<code>\lineskip</code>	32, 33
<code>\lineskiplimit</code>	33
<code>\lnot</code>	64
<code>\longdivision</code>	65, 346
<code>\longdivisionsign</code>	65
<code>\lor</code>	60
<code>\lowint</code>	344
M	
<code>\M</code>	148
<code>\m@th</code>	20, 38, 59, 77, 87, 102, 108
<code>\math</code>	71
<code>\math@bgroup</code>	34, 47, 53
<code>\math@egroup</code>	38, 54, 56
<code>\mathaccent</code>	49, 67, 97
<code>\mathaccentoverlay</code>	61, 71, 97
<code>\mathaccentwide</code>	53, 70, 97
<code>\mathalpha</code> ..	18, 41, 93, 140, 180, 184, 274
<code>\mathbacktick</code>	278
<code>\mathbf</code>	99, 104
<code>\mathbin</code>	18, 42, 78, 93, 251, 252
<code>\mathbotaccent</code>	51, 69, 97

P	
<code>\P</code>	150
<code>\peek_meaning_remove:NTF</code> ...	6, 108, 111, 114, 117, 123, 129, 135, 141, 147, 200, 205, 210, 215, 221, 227, 233
<code>\peek_N_type:TF</code>	32
<code>\phi</code>	114, 152
<code>\pkg</code>	32
<code>\plainoutput</code>	27
<code>\plainroot@</code>	47
<code>\pointint</code>	343
<code>\prg_new_conditional:Nnn</code>	24, 97, 103, 173
<code>\prg_replicate:nn</code>	13, 140
<code>\prg_return_false:</code>	29, 99, 106, 176
<code>\prg_return_true:</code>	27, 99, 106, 176
<code>\prime</code>	251
<code>\process@table</code>	114
<code>\ProcessKeysOptions</code>	218
<code>\prop_get:cnN</code>	75
<code>\prop_get:cxNTF</code>	42
<code>\prop_get:NnNTF</code>	18
<code>\prop_gput:cn</code>	33, 154
<code>\prop_gput:Nxn</code>	4, 15
<code>\prop_if_exist:cF</code>	150
<code>\prop_if_exist:cTF</code>	80
<code>\prop_item:cn</code>	156
<code>\prop_new:c</code>	31
<code>\prop_new:N</code>	101, 102
<code>\protected</code>	46–72
<code>\ProvidesExplPackage</code>	41–43
Q	
<code>\Q</code>	151
<code>\q_nil</code>	65, 68, 91, 96, 137, 149, 164
<code>\q_recursion_stop</code>	17
<code>\q_recursion_tail</code>	17
<code>\q_stop</code>	149, 151, 164, 166
<code>\q_unicode_math</code>	27, 31, 106, 107
<code>\qprime</code>	254
<code>\qqquad</code>	49
<code>\quark_if_recursion_tail_stop:n</code>	21, 22
<code>\quark_new:N</code>	107
R	
<code>\R</code>	152
<code>\r@@@t</code>	64–67, 72
<code>\raisebox</code>	79
<code>\ratio</code>	72
<code>\relax</code>	4, 5, 78, 91, 95
<code>\removenolimits</code>	21
<code>\renewcommand</code>	74
<code>\RenewNegationCommand</code>	111
<code>\RequirePackage</code>	3–7, 40
<code>\RequirePackageWithOptions</code>	54, 59
<code>\resetMathstrut@</code>	15
<code>\restore@math@cr</code>	25
<code>\restore@mathversion</code>	114
<code>\rho</code>	151
<code>\rightarrow</code>	46
<code>\rmdefault</code>	62, 117, 122
<code>\rootbox</code>	57, 89, 119
<code>\rotatebox</code>	79
<code>\rppolint</code>	342
S	
<code>\sb</code>	19
<code>\scan_stop:</code> ..	6, 26, 40, 45, 47, 48, 50, 55, 57, 59, 63, 67, 71, 75, 87, 366–410
<code>\scpolint</code>	343
<code>\scriptfont</code>	28, 31
<code>\scriptscriptfont</code>	34, 36
<code>\scriptscriptstyle</code> ..	33, 60, 67, 159, 361
<code>\scriptstyle</code> ..	27, 29, 30, 32, 39, 66, 156, 360
<code>\scshape</code>	35
<code>\selectfont</code>	99, 141
<code>\seq_clear:N</code>	64
<code>\seq_gclear:N</code>	34, 47–49, 61, 63
<code>\seq_gput_right:Nn</code>	4, 32, 50, 109
<code>\seq_gput_right:Nx</code>	82, 110, 116
<code>\seq_gremove_all:Nn</code>	129
<code>\seq_gset_eq:NN</code>	13
<code>\seq_gset_from_clist:Nn</code>	91
<code>\seq_if_empty:Nf</code>	6, 21, 30
<code>\seq_if_in:NnT</code>	122
<code>\seq_if_in:NnTF</code>	108
<code>\seq_map_break:n</code>	132
<code>\seq_map_function:NN</code>	35
<code>\seq_map_inline:Nn</code>	25, 35, 65, 126
<code>\seq_new:N</code>	86–89, 90, 99, 100
<code>\seq_put_right:Nx</code>	134
<code>\set@mathdelimiter</code>	121
<code>\set@mathaccent</code>	118
<code>\set@mathchar</code>	118
<code>\set@mathdelimiter</code>	120
<code>\set@mathsymbol</code>	119
<code>\setbox</code>	17
<code>\SetMathAlphabet</code>	22, 97–99, 103–105, 109, 110, 114, 115, 117
<code>\SetMathAlphabet@</code>	117

<code>\setmathfont</code>	3, 46, 102, 109	<code>\theta</code>	116
<code>\setmathfontface</code>	7	<code>\thinmuskip</code>	104
<code>\setmathrm</code>	24	<code>\title</code>	30
<code>\setoperatorfont</code>	12	<code>\tl_clear:N</code>	8, 61, 62
<code>\SetSymbolFont</code>		<code>\tl_const:cn</code>	101, 102
...	116, 160, 164, 197, 202, 225, 230	<code>\tl_const:Nn</code>	3, 32–37
<code>\SetSymbolFont@</code>	116	<code>\tl_gclear_new:c</code>	22
<code>\sfdefault</code>	71, 118, 123	<code>\tl_gput_right:cx</code>	272
<code>\shapedefault</code>	19,	<code>\tl_gset:Nn</code>	14, 150–152, 181–183
	22, 97, 99, 103, 104, 109, 110, 114,	<code>\tl_gset:Nx</code>	8, 143, 356
	115, 158, 161, 165, 198, 203, 226, 231	<code>\tl_if_empty:NT</code>	16, 75, 93, 122
<code>\simeq</code>	132	<code>\tl_if_empty:nTF</code>	153, 155, 157, 168, 170
<code>\skip_set:Nn</code>	27	<code>\tl_if_eq:nnF</code>	19
<code>\slash</code>	85, 86	<code>\tl_if_eq:nnT</code>	59
<code>\smallint</code>	72	<code>\tl_if_eq:nnTF</code>	107, 120
<code>\smallsetminus</code>	78	<code>\tl_if_eq:onF</code>	117–119
<code>\smash</code>	79	<code>\tl_if_eq:onT</code>	62, 71, 80, 122–124
<code>\smbllkcircle</code>	53	<code>\tl_if_exist:cF</code>	100
<code>\smwhtdiamond</code>	56	<code>\tl_if_in:NnT</code>	64, 67, 84
<code>\sp</code>	8	<code>\tl_if_in:nnT</code>	17
<code>\space</code>	37, 136	<code>\tl_if_in:NnTF</code>	89
<code>\sqint</code>	343	<code>\tl_if_in:VnTF</code>	106
<code>\sqrt</code>	346	<code>\tl_if_single_p:n</code>	103
<code>\sqrtsign</code>	103	<code>\tl_map_inline:nn</code>	108
<code>\std@equal</code>	5, 10	<code>\tl_new:N</code>	38–71, 72, 73
<code>\std@minus</code>	4, 9	<code>\tl_put_left:Nn</code>	4
<code>\sterling</code>	55	<code>\tl_put_right:Nn</code>	5, 19
<code>\str_if_eq:eeT</code>	162, 200, 228	<code>\tl_put_right:NV</code>	46
<code>\str_if_eq:eeTF</code>	90	<code>\tl_remove_all:Nn</code>	23, 57, 71, 72
<code>\string</code>	66–72	<code>\tl_remove_once:Nn</code>	2, 125
<code>\subarray</code>	20	<code>\tl_set:cx</code>	14–16
<code>\sumint</code>	341	<code>\tl_set:Nn</code>	6,
<code>\sym</code>	72		7, 9, 12, 18, 18, 20, 45, 50, 52, 53,
<code>\symit</code>	16		53, 55, 56, 56, 60, 87, 163, 338, 346
<code>\symrm</code>	87	<code>\tl_set:No</code>	37, 67, 69
<code>\symup</code>	15, 87	<code>\tl_set:Nx</code>	41, 41, 70, 87, 92,
<code>\sys_if_engine_luatex:T</code>	52		93, 94, 95, 98, 99, 112, 114, 121, 124
<code>\sys_if_engine_xetex:T</code>	57	<code>\tl_set_eq:NN</code>	57, 58, 76
T			
<code>\tex_fam:D</code>	87	<code>\tl_to_str:N</code>	70
<code>\tex_global:D</code>	34, 71, 75	<code>\tl_to_str:n</code>	41, 83–85, 112
<code>\tex_iffontchar:D</code>	26	<code>\tl_trim_spaces:N</code>	73
<code>\tex_let:D</code>	34	<code>\tl_trim_spaces:n</code>	93, 94, 98, 99, 114, 135
<code>\tex_mathcode:D</code>	71, 75	<code>\tl_use:c</code>	23, 27, 137
<code>\textfont</code>	16, 19, 22, 25	<code>\tl_use:N</code>	4, 351
<code>\textstyle</code>	21, 65, 72, 359	<code>\to</code>	46
<code>\tfrac</code>	21, 217	<code>\token_case_meaning:Nn</code>	25, 38
<code>\the</code>	91, 366–410	<code>\token_if_cs_p:N</code>	103
		<code>\token_if_eq_meaning:NNT</code>	36

<code>\varointclockwise</code>	341		
<code>\vcenter</code>	22		
<code>\vee</code>	60		
<code>\version@elt</code>	113		
<code>\version@list</code>	113		
<code>\Vert</code>	82		
<code>\vrule</code>	47, 55		
<code>\vysmwhtcircle</code>	52		
		X	
		<code>\XeTeXradical</code>	23
		Y	
		<code>\yen</code>	54
		Z	
		<code>\Z</code>	153
		<code>\z@</code>	17, 18, 47, 55
	W		
<code>\wedge</code>	59		