

# Experimental Unicode mathematical typesetting: The unicode-math package

WILL ROBERTSON

*Philipp Stephani, Joseph Wright, Khaled Hosny, and others*

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

2019/03/04 0.8o

## Contents

<b>I</b>	<b>unicode-math.dtx</b>	<b>6</b>
1	Package metadata	6
2	The unicode-math.sty loading file	7
<b>II</b>	<b>um-code-opening.dtx</b>	<b>8</b>
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
<b>III</b>	<b>um-code-variables.dtx</b>	<b>14</b>
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
<b>IV</b>	<b>um-code-api.dtx</b>	<b>18</b>
5	Programmers' interface	18
<b>V</b>	<b>um-code-ui.dtx</b>	<b>19</b>
6	The user interface commands	19
<b>VI</b>	<b>um-code-pkgopt.dtx</b>	<b>21</b>
7	setup and package options	21
7.1	Defaults . . . . .	26
<b>VII</b>	<b>um-code-msg.dtx</b>	<b>27</b>
8	Error messages	27
<b>VIII</b>	<b>um-code-usv.dtx</b>	<b>30</b>
9	Alphabet Unicode positions	30
9.1	STIX fonts . . . . .	36
<b>IX</b>	<b>um-code-setchar.dtx</b>	<b>40</b>
10	Setting up maths chars	40
10.1	A token list to contain the data of the math table . . . . .	40
10.2	Definitions of the active math characters . . . . .	40
10.3	Commands for each symbol/glyph/char . . . . .	41
<b>X</b>	<b>um-code-mathtext.dtx</b>	<b>45</b>

<b>11 Maths text commands</b>	<b>45</b>
11.1 <code>\setmathfontface</code> . . . . .	45
11.2 Hooks into $\LaTeX 2_{\epsilon}$ . . . . .	45
11.3 Hooks into <code>fontspec</code> . . . . .	47
11.3.1 Text font . . . . .	47
11.3.2 Maths font . . . . .	48
<b>XI <code>um-code-main.dtx</code></b>	<b>50</b>
<b>12 The main <code>\setmathfont</code> macro</b>	<b>50</b>
12.1 Functions for setting up symbols with <code>mathcodes</code> . . . . .	56
12.2 Active math characters . . . . .	57
12.3 Delimiter codes . . . . .	58
12.4 (Big) operators . . . . .	59
12.5 Radicals . . . . .	59
12.6 <code>Fontdimens</code> . . . . .	59
<b>XII <code>um-code-fontopt.dtx</code></b>	<b>62</b>
<b>13 Font loading options</b>	<b>62</b>
13.1 Math version . . . . .	62
13.2 Script and <code>scriptscript</code> font options . . . . .	62
13.3 Range processing . . . . .	62
<b>XIII <code>um-code-fontparam.dtx</code></b>	<b>68</b>
<b>14 Cross-platform interface for font parameters</b>	<b>68</b>
14.1 Historical commands . . . . .	73
<b>XIV <code>um-code-mathmap.dtx</code></b>	<b>74</b>
<b>15 Defining the math alphabets per style</b>	<b>74</b>
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
<b>XV <code>um-code-sym-commands.dtx</code></b>	<b>84</b>

<b>16 Mapping in maths alphabets</b>	<b>84</b>
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
<b>XVI um-code-alphabets.dtx</b>	<b>88</b>
<b>17 Setting up alphabets</b>	<b>88</b>
17.1 Upright: up . . . . .	88
17.2 Italic: it . . . . .	89
17.3 Blackboard or double-struck: bb and bbit . . . . .	91
17.4 Script and caligraphic: 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: bfsfup . . . . .	100
17.14 Bold italic sans serif: bfsfit . . . . .	103
<b>XVII um-code-primes.dtx</b>	<b>106</b>
<b>18 Primes</b>	<b>106</b>
<b>XVIII um-code-sscript.dtx</b>	<b>113</b>
<b>19 Unicode sub- and super-scripts</b>	<b>113</b>
<b>XIX um-code-compat.dtx</b>	<b>117</b>
<b>20 Compatibility</b>	<b>117</b>
<b>21 Patching/augmenting 3rd-party packages</b>	<b>118</b>
21.1 url . . . . .	118
21.2 mathtools . . . . .	118
21.3 colonequals . . . . .	120

<b>XX</b>	<b>um-code-amsmath.dtx</b>	<b>121</b>
<b>22</b>	<b>Compatibility with amsmath</b>	<b>121</b>
<b>XXI</b>	<b>um-code-epilogue.dtx</b>	<b>125</b>
<b>23</b>	<b>Epilogue</b>	<b>125</b>
23.1	Resolving Greek symbol name control sequences . . . . .	125
23.2	Unicode radicals . . . . .	125
23.2.1	Active fractions . . . . .	126
23.3	Synonyms and all the rest . . . . .	127
23.3.1	\not . . . . .	129
23.3.2	Full-width remapping . . . . .	130
23.4	Legacy characters . . . . .	130
<b>24</b>	<b>A secret hook</b>	<b>131</b>

## 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> {2019/03/04} {0.8o} {Unicode maths in XeLaTeX and LuaLaTeX}

```

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

```

45 <*dtx>
46 \date{
47   \def\filedate{2019/03/04}
48   \def\fileversion{0.8o}
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 { \RequirePackageWithOptions{unicode-math-luatex} }
53 <base>\sys_if_engine_xetex:T { \RequirePackageWithOptions{unicode-math-xetex} }
54 <base>\sys_if_engine_pdftex:T
55 <base> {
56 <base>   \msg_new:nnn {unicode-math} {not-pdftex}
57 <base>     { Cannot~ be~ run~ with~ pdfLaTeX!\! Use~ XeLaTeX~ or~ LuaLa-
58 <base>       TeX~ instead. }
59 <base> }
60 <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{filehook}
6 \RequirePackage{fix-cm}
7 <LU>\RequirePackage{lualatex-math}
```

#### 3.1 *expl3 variants*

Variants needed from expl3:

```
8 \cs_set_protected_nopar:Npn \exp_last_unbraced:NNx { \::N \::x_unbraced \::: }
   For fontspec:
9 \cs_generate_variant:Nn \fontspec_set_family:Nnn {Nx,Nxx}
10 \cs_generate_variant:Nn \prop_get:NnNTF {cx}
11 \cs_generate_variant:Nn \tl_if_eq:nnF {o}
```

#### 3.2 *Low level commands*

```
12 \cs_set_eq:NN \@@_group_begin: \group_begin:
13 \cs_set_protected:Npn \@@_group_end:n #1 { #1 \group_end: }
14 \cs_set_eq:NN \@@_group_begin_frozen: \@@_group_begin:
15 \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
```

```
16 \prg_new_conditional:Nnn \@@_glyph_if_exist:Nn {p,TF,T,F}
17 {
18   \tex_iffontchar:D #1 #2 \scan_stop:
19   \prg_return_true:
20   \else:
21   \prg_return_false:
22   \fi:
23 }
```



\@@\_fontface\_gset\_eq:NN

```
24 \cs_set_protected:Nn \@@_fontface_gset_eq:NN
25 {
26   \tex_global:D \tex_let:D #1 #2
27 }
28 \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
29 \cs_set:Npn \@@_set_mathcode:nnnn #1#2#3#4
30 {
31   \Umathcode \int_eval:n {#1} =
32   \mathchar@type#2 \csname sym#3\endcsname \int_eval:n {#4} \scan_stop:
33 }
34 \cs_set:Npn \@@_set_mathcode:nnn #1#2#3
35 {
36   \Umathcode \int_eval:n {#1} =
37   \mathchar@type#2 \csname sym#3\endcsname \int_eval:n {#1} \scan_stop:
38 }
```

\@@\_set\_mathchar:NNnn

```
\@@_set_mathchar:cNnn
39 \cs_set:Npn \@@_set_mathchar:NNnn #1#2#3#4
40 {
41   \Umathchardef #1 =
42   \mathchar@type#2 \csname sym#3\endcsname \int_eval:n {#4} \scan_stop:
43 }
44 \cs_generate_variant:Nn \@@_set_mathchar:NNnn {c}
```

\@@\_set\_delcode:nnn

```
45 \cs_new:Nn \@@_set_delcode:nnn
46 {
47   \Udelcode#2 = \csname sym#1\endcsname #3 \scan_stop:
48 }
```

\@@\_radical:nn

```
49 \cs_new:Nn \@@_radical:nn
50 {
51   \Uradical \csname sym#1\endcsname #2 \scan_stop:
52 }
```

\@@\_delimiter:Nnn

```
53 \cs_new:Nn \@@_delimiter:Nnn
54 {
55   \Udelimiter \mathchar@type#1 \csname sym#2\endcsname #3 \scan_stop:
56 }
```

`\@@_accent:nnn`

```
57 \cs_new:Nn \@@_accent:nnn
58 {
59   \Umathaccent #1~ \mathchar@type\mathaccent \use:c { sym #2 } #3 \scan_stop:
60 }
```

`\@@_char_gmake_mathactive:N`

`\@@_char_gmake_mathactive:n`

```
61 \cs_new:Nn \@@_char_gmake_mathactive:N
62 {
63   \tex_global:D \tex_mathcode:D `#1 = "8000 \scan_stop:
64 }
65 \cs_new:Nn \@@_char_gmake_mathactive:n
66 {
67   \tex_global:D \tex_mathcode:D \int_eval:n {#1} = "8000 \scan_stop:
68 }
```

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

```
69 \cs_new:Nn \@@_mathactive_remap:nn
70 {
71   \group_begin:
72     \cs_set_protected:Npn \@@_tmp: {#2}
73     \@@_char_gmake_mathactive:n {#1}
74     \char_gset_active_eq:nN {#1} \@@_tmp:
75   \group_end:
76 }
```

### 3.3.2 NFSS-related interfaces

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

```
77 \cs_new_protected:Nn \@@_mathgroup_set:n
78 {
79   \tex_fam:D #1 \scan_stop:
80 }
```

### 3.3.3 Font parameters

`\@@_copy_fontdimen:nnN`

```
81 \cs_new:Nn \@@_copy_fontdimen:nnN
82 {
83   \fontdimen #1 \font = \the \fontdimen #2 #3 \relax
84 }
```

`\@@_zero_fontdimen:n`

```
85 \cs_new:Nn \@@_zero_fontdimen:n
86 {
87   \fontdimen #1 \font = 0pt\relax
88 }
```

`\@@_fontdimen_from_param:Nnn` This function extracts the math font dimen #3 from the font #1 and sets fontdimen #2 of the same font to that value.

Use Xe<sub>La</sub>TeX's fontdimen approach because it's tidy. We don't need bells and whistles here.

```

89 <*LU>
90 \cs_new_protected:Nn \@@_fontdimen_from_param:nn
91   {
92     \fontdimen #1 \font =
93     \lua_now:n { fontspect.mathfontdimen(font.current(),"#2") }
94     \scan_stop:
95   }
96 </LU>

```

`\@@_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.<sup>1</sup>

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

```

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.)

```

108 \tl_map_inline:nn
109   {
110     \new@mathgroup\cdp@list\cdp@elt\DeclareMathSizes
111     \DeclareMathSizes\newmathalphabet\newmathalphabet@@\newmathalphabet@@@
112     \DeclareMathVersion\define@mathalphabet\define@mathgroup\addtoversion
113     \version@list\version@elt\alpha@list\alpha@elt
114     \restore@mathversion\init@restore@version\dorestore@version\process@table
115     \new@mathversion\DeclareSymbolFont\group@list\group@elt

```

<sup>1</sup>'u.s.v.' stands for 'Unicode scalar value'.

```

116 \new@symbolfont\SetSymbolFont\SetSymbolFont@get@cdp
117 \DeclareMathAlphabet\new@mathalphabet\SetMathAlphabet\SetMathAlphabet@
118 \DeclareMathAccent\set@mathaccent\DeclareMathSymbol\set@mathchar
119 \set@mathsymbol\DeclareMathDelimiter\@xxDeclareMathDelimiter
120 \@DeclareMathDelimiter\@xDeclareMathDelimiter\set@mathdelimiter
121 \set@@mathdelimiter\DeclareMathRadical\mathchar@type
122 \DeclareSymbolFontAlphabet\DeclareSymbolFontAlphabet@
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 {nnxx}
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*

```
32 \tl_if_exist:NF \g_@@_secret_hook_tl { \tl_new:N \g_@@_secret_hook_tl }
```

For displaying in warning messages, etc.:

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

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

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

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

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

```
71 \tl_new:N \g_@@_sqrt_font_cmd_tl
72 \tl_new:N \g_@@_prime_font_cmd_tl
```

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

```
73 (*LU)
74 \tl_new:N \g_@@_mathparam_store_tl
75 </LU>
```

#### 4.4 *clist*

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

#### 4.5 *seq*

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

`\g_@@_mathclasses_seq` Every math class.

```
91 \seq_new:N \g_@@_mathclasses_seq
92 \seq_gset_from_clist:Nn \g_@@_mathclasses_seq
93 {
94   \mathord, \mathalpha, \mathbin, \mathrel, \mathpunct,
95   \mathop,
96   \mathopen, \mathclose,
97   \mathfence, \mathover, \mathunder,
98   \mathaccent, \mathaccentoverlay, \mathbotaccent, \mathaccentwide, \mathbotaccentwide
99 }
```

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

```
100 \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 `\symbfit` 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!

```
101 \seq_new:N \g_@@_mathstyles_seq
```



## 4.6 *prop*

102 \prop\_new:N \g\_@@\_supers\_prop

103 \prop\_new:N \g\_@@\_subs\_prop

## 4.7 *muskip*

104 \muskip\_new:N \g\_@@\_primekern\_muskip

105 \muskip\_gset:Nn \g\_@@\_primekern\_muskip { -\thinmuskip/2 }% arbitrary

## 4.8 *fp*

106 \fp\_new:N \g\_@@\_size\_tfsf\_fp

107 \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.

108 \quark\_new:N \q\_unicode\_math

109 </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<sup>A</sup>T<sub>E</sub>X'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         \cs_if_exist:NTF \tfrac
190         { \bool_set_true:N \l_@@_smallfrac_bool }
191         {
192             \@@_warning:n {no-tfrac}
193             \bool_set_false:N \l_@@_smallfrac_bool
194         }
195         \use:c {@@_setup_active_frac:}
196     }
197
198     {normalsize}
199     {
200         \bool_set_false:N \l_@@_smallfrac_bool
201         \use:c {@@_setup_active_frac:}
202     }
203 }

```

### *Debug/tracing*

```

204 \keys_define:nn {unicode-math}
205 {
206     warnings-off .code:n =
207     {
208         \clist_map_inline:nn {#1}
209         { \msg_redirect_name:nnn { unicode-math } { ##1 } { none } }
210     }
211 }

```

```

212 \@_keys_choices:nn {trace}
213   {
214     {on}   {} % default
215     {debug} { \msg_redirect_module:nnn { unicode-math } { log } { warning } }
216     {off}   { \msg_redirect_module:nnn { unicode-math } { log } { none } }
217   }

```

## 7.1 Defaults

```

218 \unimathsetup {math-style=TeX}
219 \unimathsetup {slash-delimiter=ascii}
220 \unimathsetup {trace=off}
221 \unimathsetup {mathrm=text,mathit=text,mathbf=text,mathsf=text,mathtt=text}
222 \cs_if_exist:NT \frac { \unimathsetup {active-frac=small} }
223 \ProcessKeysOptions {unicode-math}
224 </package>

```

## File VII

# um-code-msg.dtx

### 8 *Error messages*

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

```

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

```

```

90 {
91   Command `#1` in math range is not recognised as a maths symbol.
92   Check file "unicode-math-table.tex" for allowable commands.
93 }
94 \@@_msg_new:nn {legacy-char-not-supported}
95 {
96   Command `#1` is a legacy maths symbol that is not supported by unicode-
math.
97 }
98 \@@_msg_new:nn {range-not-bf-sf}
99 {
100   Range alphabets cannot include alphabets referring to `bf`, `sf`, or `bfsf`
101   since they relate to input commands not output glyphs.
102   Use `bfit` or `bfup` (etc.) to specify which.
103 }
104 \@@_msg_new:nn {no-main-font}
105 {
106   No main maths font has been set up yet.\\If you simply want 'the de-
fault', use: \\
107   \iow_indent:n {\token_to_str:N\setmathfont{latinmodern-math.otf}}
108 }
109 \@@_msg_new:nn {not-ot-math}
110 {
111   The first font loaded by unicode-math must be an OpenType Math font (with script=math).
112   If you simply want 'the default' before loading supplemen-
tary fonts over the top for certain
113   ranges, use: \\
114   \iow_indent:n {\token_to_str:N\setmathfont{latinmodern-math.otf}}
115 }
116 \char_set_catcode_ignore:n {32}
117 </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.<sup>2</sup>

```
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 {frak} {Latin} {"1D504}
32 \usv_set:nnn {frak} {latin} {"1D51E}
33 \usv_set:nnn {sf}   {num}   {"1D7E2}
34 \usv_set:nnn {sfup} {num}   {"1D7E2}
```

---

<sup>2</sup>'u.s.v.' stands for 'Unicode scalar value'.

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

**Bold weights:**

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

**The 'auto' bolds:**

72 \usv\_set:nnn {bfsf} {Latin} { \bool\_if:NTF \g\_@@\_upLatin\_bool \g\_@@\_bfsfup\_Latin\_usv \g\_@@\_bfsfi  
 73 \usv\_set:nnn {bfsf} {latin} { \bool\_if:NTF \g\_@@\_uplatin\_bool \g\_@@\_bfsfup\_latin\_usv \g\_@@\_bfsfi  
 74 \usv\_set:nnn {bfsf} {Greek} { \bool\_if:NTF \g\_@@\_upGreek\_bool \g\_@@\_bfsfup\_Greek\_usv \g\_@@\_bfsfi  
 75 \usv\_set:nnn {bfsf} {greek} { \bool\_if:NTF \g\_@@\_upgreek\_bool \g\_@@\_bfsfup\_greek\_usv \g\_@@\_bfsfi  
 76 \usv\_set:nnn {bf} {Latin} { \bool\_if:NTF \g\_@@\_bfupLatin\_bool \g\_@@\_bfup\_Latin\_usv \g\_@@\_bfit\_L  
 77 \usv\_set:nnn {bf} {latin} { \bool\_if:NTF \g\_@@\_bfuplatin\_bool \g\_@@\_bfup\_latin\_usv \g\_@@\_bfit\_L  
 78 \usv\_set:nnn {bf} {Greek} { \bool\_if:NTF \g\_@@\_bfupGreek\_bool \g\_@@\_bfup\_Greek\_usv \g\_@@\_bfit\_G  
 79 \usv\_set:nnn {bf} {greek} { \bool\_if:NTF \g\_@@\_bfupgreek\_bool \g\_@@\_bfup\_greek\_usv \g\_@@\_bfit\_G

*Greek variants* Upright:

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

**Bold:**

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

*Italic:*

98 \usv\_set:nnn {it} {\varTheta} {"1D6F3}  
99 \usv\_set:nnn {it} {\epsilon} {"1D716}  
100 \usv\_set:nnn {it} {\vartheta} {"1D717}  
101 \usv\_set:nnn {it} {\varkappa} {"1D718}  
102 \usv\_set:nnn {it} {\phi} {"1D719}  
103 \usv\_set:nnn {it} {\varrho} {"1D71A}  
104 \usv\_set:nnn {it} {\varpi} {"1D71B}

**Bold italic:**

105 \usv\_set:nnn {bfit} {\varTheta} {"1D72D}  
106 \usv\_set:nnn {bfit} {\epsilon} {"1D750}  
107 \usv\_set:nnn {bfit} {\vartheta} {"1D751}  
108 \usv\_set:nnn {bfit} {\varkappa} {"1D752}  
109 \usv\_set:nnn {bfit} {\phi} {"1D753}  
110 \usv\_set:nnn {bfit} {\varrho} {"1D754}  
111 \usv\_set:nnn {bfit} {\varpi} {"1D755}

**Bold sans:**

112 \usv\_set:nnn {bfsfup} {\varTheta} {"1D767}  
113 \usv\_set:nnn {bfsfup} {\epsilon} {"1D78A}  
114 \usv\_set:nnn {bfsfup} {\vartheta} {"1D78B}  
115 \usv\_set:nnn {bfsfup} {\varkappa} {"1D78C}  
116 \usv\_set:nnn {bfsfup} {\phi} {"1D78D}  
117 \usv\_set:nnn {bfsfup} {\varrho} {"1D78E}  
118 \usv\_set:nnn {bfsfup} {\varpi} {"1D78F}

**Bold sans italic:**

119 \usv\_set:nnn {bfsfit} {\varTheta} {"1D7A1}



120 \usv\_set:nnn {bfsfit} {epsilon} {"1D7C4}  
 121 \usv\_set:nnn {bfsfit} {vartheta} {"1D7C5}  
 122 \usv\_set:nnn {bfsfit} {varkappa} {"1D7C6}  
 123 \usv\_set:nnn {bfsfit} {phi} {"1D7C7}  
 124 \usv\_set:nnn {bfsfit} {varrho} {"1D7C8}  
 125 \usv\_set:nnn {bfsfit} {varpi} {"1D7C9}

**Nabla:**

126 \usv\_set:nnn {up} {Nabla} {"02207}  
 127 \usv\_set:nnn {it} {Nabla} {"1D6FB}  
 128 \usv\_set:nnn {bfup} {Nabla} {"1D6C1}  
 129 \usv\_set:nnn {bfit} {Nabla} {"1D735}  
 130 \usv\_set:nnn {bfsfup} {Nabla} {"1D76F}  
 131 \usv\_set:nnn {bfsfit} {Nabla} {"1D7A9}

**Partial:**

132 \usv\_set:nnn {up} {partial} {"02202}  
 133 \usv\_set:nnn {it} {partial} {"1D715}  
 134 \usv\_set:nnn {bfup} {partial} {"1D6DB}  
 135 \usv\_set:nnn {bfit} {partial} {"1D74F}  
 136 \usv\_set:nnn {bfsfup} {partial} {"1D789}  
 137 \usv\_set:nnn {bfsfit} {partial} {"1D7C3}

*Exceptions* Upright uppercase:

138 \usv\_set:nnn {up} {B} {"\B}  
 139 \usv\_set:nnn {up} {C} {"\C}  
 140 \usv\_set:nnn {up} {D} {"\D}  
 141 \usv\_set:nnn {up} {E} {"\E}  
 142 \usv\_set:nnn {up} {F} {"\F}  
 143 \usv\_set:nnn {up} {H} {"\H}  
 144 \usv\_set:nnn {up} {I} {"\I}  
 145 \usv\_set:nnn {up} {L} {"\L}  
 146 \usv\_set:nnn {up} {M} {"\M}  
 147 \usv\_set:nnn {up} {N} {"\N}  
 148 \usv\_set:nnn {up} {P} {"\P}  
 149 \usv\_set:nnn {up} {Q} {"\Q}  
 150 \usv\_set:nnn {up} {R} {"\R}  
 151 \usv\_set:nnn {up} {Z} {"\Z}

**Italic uppercase:**

152 \usv\_set:nnn {it} {B} {"1D435}  
 153 \usv\_set:nnn {it} {C} {"1D436}  
 154 \usv\_set:nnn {it} {D} {"1D437}  
 155 \usv\_set:nnn {it} {E} {"1D438}  
 156 \usv\_set:nnn {it} {F} {"1D439}  
 157 \usv\_set:nnn {it} {H} {"1D43B}  
 158 \usv\_set:nnn {it} {I} {"1D43C}  
 159 \usv\_set:nnn {it} {L} {"1D43F}  
 160 \usv\_set:nnn {it} {M} {"1D440}  
 161 \usv\_set:nnn {it} {N} {"1D441}

162 \usv\_set:nnn {it} {P} {"1D443}  
 163 \usv\_set:nnn {it} {Q} {"1D444}  
 164 \usv\_set:nnn {it} {R} {"1D445}  
 165 \usv\_set:nnn {it} {Z} {"1D44D}

Upright lowercase (needed for later mappings):

166 \usv\_set:nnn {up} {d} {\`d}  
 167 \usv\_set:nnn {up} {e} {\`e}  
 168 \usv\_set:nnn {up} {g} {\`g}  
 169 \usv\_set:nnn {up} {h} {\`h}  
 170 \usv\_set:nnn {up} {i} {\`i}  
 171 \usv\_set:nnn {up} {j} {\`j}  
 172 \usv\_set:nnn {up} {o} {\`o}

Italic lowercase:

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

Latin 'h':

180 \usv\_set:nnn {bb} {h} {"1D559}  
 181 \usv\_set:nnn {tt} {h} {"1D691}  
 182 \usv\_set:nnn {scr} {h} {"1D4BD}  
 183 \usv\_set:nnn {frak} {h} {"1D525}  
 184 \usv\_set:nnn {bfup} {h} {"1D421}  
 185 \usv\_set:nnn {bfit} {h} {"1D489}  
 186 \usv\_set:nnn {sfup} {h} {"1D5C1}  
 187 \usv\_set:nnn {sfit} {h} {"1D629}  
 188 \usv\_set:nnn {bffrak} {h} {"1D58D}  
 189 \usv\_set:nnn {bfscr} {h} {"1D4F1}  
 190 \usv\_set:nnn {bfsfup} {h} {"1D5F5}  
 191 \usv\_set:nnn {bfsfit} {h} {"1D65D}

Dotless 'i' and 'j':

192 \usv\_set:nnn {up} {dotlessi} {"00131}  
 193 \usv\_set:nnn {up} {dotlessj} {"00237}  
 194 \usv\_set:nnn {it} {dotlessi} {"1D6A4}  
 195 \usv\_set:nnn {it} {dotlessj} {"1D6A5}

Blackboard:

196 \usv\_set:nnn {bb} {C} {"2102}  
 197 \usv\_set:nnn {bb} {H} {"210D}  
 198 \usv\_set:nnn {bb} {N} {"2115}  
 199 \usv\_set:nnn {bb} {P} {"2119}  
 200 \usv\_set:nnn {bb} {Q} {"211A}  
 201 \usv\_set:nnn {bb} {R} {"211D}  
 202 \usv\_set:nnn {bb} {Z} {"2124}

203 \usv\_set:nnn {up} {Pi} {"003A0}  
 204 \usv\_set:nnn {up} {pi} {"003C0}  
 205 \usv\_set:nnn {up} {Gamma} {"00393}  
 206 \usv\_set:nnn {up} {gamma} {"003B3}  
 207 \usv\_set:nnn {up} {summation} {"02211}  
 208 \usv\_set:nnn {it} {Pi} {"1D6F1}  
 209 \usv\_set:nnn {it} {pi} {"1D70B}  
 210 \usv\_set:nnn {it} {Gamma} {"1D6E4}  
 211 \usv\_set:nnn {it} {gamma} {"1D6FE}  
 212 \usv\_set:nnn {bb} {Pi} {"0213F}  
 213 \usv\_set:nnn {bb} {pi} {"0213C}  
 214 \usv\_set:nnn {bb} {Gamma} {"0213E}  
 215 \usv\_set:nnn {bb} {gamma} {"0213D}  
 216 \usv\_set:nnn {bb} {summation} {"02140}

**Italic blackboard:**

217 \usv\_set:nnn {bbit} {D} {"2145}  
 218 \usv\_set:nnn {bbit} {d} {"2146}  
 219 \usv\_set:nnn {bbit} {e} {"2147}  
 220 \usv\_set:nnn {bbit} {i} {"2148}  
 221 \usv\_set:nnn {bbit} {j} {"2149}

**Script:**

222 \usv\_set:nnn {scr} {B} {"212C}  
 223 \usv\_set:nnn {scr} {E} {"2130}  
 224 \usv\_set:nnn {scr} {F} {"2131}  
 225 \usv\_set:nnn {scr} {H} {"210B}  
 226 \usv\_set:nnn {scr} {I} {"2110}  
 227 \usv\_set:nnn {scr} {L} {"2112}  
 228 \usv\_set:nnn {scr} {M} {"2133}  
 229 \usv\_set:nnn {scr} {R} {"211B}  
 230 \usv\_set:nnn {scr} {e} {"212F}  
 231 \usv\_set:nnn {scr} {g} {"210A}  
 232 \usv\_set:nnn {scr} {o} {"2134}

**Caligraphic:**

233 \usv\_set:nnn {cal} {B} {"212C}  
 234 \usv\_set:nnn {cal} {E} {"2130}  
 235 \usv\_set:nnn {cal} {F} {"2131}  
 236 \usv\_set:nnn {cal} {H} {"210B}  
 237 \usv\_set:nnn {cal} {I} {"2110}  
 238 \usv\_set:nnn {cal} {L} {"2112}  
 239 \usv\_set:nnn {cal} {M} {"2133}  
 240 \usv\_set:nnn {cal} {R} {"211B}

**Fraktur:**

241 \usv\_set:nnn {frak} {C} {"212D}  
 242 \usv\_set:nnn {frak} {H} {"210C}  
 243 \usv\_set:nnn {frak} {I} {"2111}  
 244 \usv\_set:nnn {frak} {R} {"211C}  
 245 \usv\_set:nnn {frak} {Z} {"2128}

246 `</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.

247 `<*stix>`

### *Upright*

```
248 \usv_set:nnn {stixsfup}{partial}{E17C}
249 \usv_set:nnn {stixsfup}{Greek}{E17D}
250 \usv_set:nnn {stixsfup}{greek}{E196}
251 \usv_set:nnn {stixsfup}{varTheta}{E18E}
252 \usv_set:nnn {stixsfup}{epsilon}{E1AF}
253 \usv_set:nnn {stixsfup}{vartheta}{E1B0}
254 \usv_set:nnn {stixsfup}{varkappa}{0000} % ???
255 \usv_set:nnn {stixsfup}{phi}{E1B1}
256 \usv_set:nnn {stixsfup}{varrho}{E1B2}
257 \usv_set:nnn {stixsfup}{varpi}{E1B3}
258 \usv_set:nnn {stixupslash}{Greek}{E2FC}
```

### *Italic*

```
259 \usv_set:nnn {stixbbit}{A}{E154}
260 \usv_set:nnn {stixbbit}{B}{E155}
261 \usv_set:nnn {stixbbit}{E}{E156}
262 \usv_set:nnn {stixbbit}{F}{E157}
263 \usv_set:nnn {stixbbit}{G}{E158}
264 \usv_set:nnn {stixbbit}{I}{E159}
265 \usv_set:nnn {stixbbit}{J}{E15A}
266 \usv_set:nnn {stixbbit}{K}{E15B}
267 \usv_set:nnn {stixbbit}{L}{E15C}
268 \usv_set:nnn {stixbbit}{M}{E15D}
269 \usv_set:nnn {stixbbit}{O}{E15E}
270 \usv_set:nnn {stixbbit}{S}{E15F}
271 \usv_set:nnn {stixbbit}{T}{E160}
272 \usv_set:nnn {stixbbit}{U}{E161}
273 \usv_set:nnn {stixbbit}{V}{E162}
274 \usv_set:nnn {stixbbit}{W}{E163}
275 \usv_set:nnn {stixbbit}{X}{E164}
276 \usv_set:nnn {stixbbit}{Y}{E165}

277 \usv_set:nnn {stixbbit}{a}{E166}
278 \usv_set:nnn {stixbbit}{b}{E167}
279 \usv_set:nnn {stixbbit}{c}{E168}
280 \usv_set:nnn {stixbbit}{f}{E169}
281 \usv_set:nnn {stixbbit}{g}{E16A}
```

282 \usv\_set:nnn {stixbbit}{h}{E16B}  
283 \usv\_set:nnn {stixbbit}{k}{E16C}  
284 \usv\_set:nnn {stixbbit}{l}{E16D}  
285 \usv\_set:nnn {stixbbit}{m}{E16E}  
286 \usv\_set:nnn {stixbbit}{n}{E16F}  
287 \usv\_set:nnn {stixbbit}{o}{E170}  
288 \usv\_set:nnn {stixbbit}{p}{E171}  
289 \usv\_set:nnn {stixbbit}{q}{E172}  
290 \usv\_set:nnn {stixbbit}{r}{E173}  
291 \usv\_set:nnn {stixbbit}{s}{E174}  
292 \usv\_set:nnn {stixbbit}{t}{E175}  
293 \usv\_set:nnn {stixbbit}{u}{E176}  
294 \usv\_set:nnn {stixbbit}{v}{E177}  
295 \usv\_set:nnn {stixbbit}{w}{E178}  
296 \usv\_set:nnn {stixbbit}{x}{E179}  
297 \usv\_set:nnn {stixbbit}{y}{E17A}  
298 \usv\_set:nnn {stixbbit}{z}{E17B}

299 \usv\_set:nnn {stixsfit}{Numerals}{E1B4}  
300 \usv\_set:nnn {stixsfit}{partial}{E1BE}  
301 \usv\_set:nnn {stixsfit}{Greek}{E1BF}  
302 \usv\_set:nnn {stixsfit}{greek}{E1D8}  
303 \usv\_set:nnn {stixsfit}{varTheta}{E1D0}  
304 \usv\_set:nnn {stixsfit}{epsilon}{E1F1}  
305 \usv\_set:nnn {stixsfit}{vartheta}{E1F2}  
306 \usv\_set:nnn {stixsfit}{varkappa}{0000} % ???  
307 \usv\_set:nnn {stixsfit}{phi}{E1F3}  
308 \usv\_set:nnn {stixsfit}{varrho}{E1F4}  
309 \usv\_set:nnn {stixsfit}{varpi}{E1F5}

310 \usv\_set:nnn {stixcal}{Latin}{E22D}  
311 \usv\_set:nnn {stixcal}{num}{E262}  
312 \usv\_set:nnn {scr}{num}{48}  
313 \usv\_set:nnn {it}{num}{48}

314 \usv\_set:nnn {stixsfitslash}{Latin}{E294}  
315 \usv\_set:nnn {stixsfitslash}{latin}{E2C8}  
316 \usv\_set:nnn {stixsfitslash}{greek}{E32C}  
317 \usv\_set:nnn {stixsfitslash}{epsilon}{E37A}  
318 \usv\_set:nnn {stixsfitslash}{vartheta}{E35E}  
319 \usv\_set:nnn {stixsfitslash}{varkappa}{E374}  
320 \usv\_set:nnn {stixsfitslash}{phi}{E360}  
321 \usv\_set:nnn {stixsfitslash}{varrho}{E376}  
322 \usv\_set:nnn {stixsfitslash}{varpi}{E362}  
323 \usv\_set:nnn {stixsfitslash}{digamma}{E36A}

*Bold*

324 \usv\_set:nnn {stixbfupslash}{Greek}{E2FD}  
325 \usv\_set:nnn {stixbfupslash}{Digamma}{E369}  
326 \usv\_set:nnn {stixbfbb}{A}{E38A}

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

*Bold Italic*

367 \usv\_set:nnn {stixbfsfit}{Numerals}{E1F6}  
 368 \usv\_set:nnn {stixfbfbit}{A}{E200}  
 369 \usv\_set:nnn {stixfbfbit}{B}{E201}  
 370 \usv\_set:nnn {stixfbfbit}{E}{E203}  
 371 \usv\_set:nnn {stixfbfbit}{F}{E204}  
 372 \usv\_set:nnn {stixfbfbit}{G}{E205}

373 \usv\_set:nnn {stixfbbit}{I}{E206}  
374 \usv\_set:nnn {stixfbbit}{J}{E207}  
375 \usv\_set:nnn {stixfbbit}{K}{E208}  
376 \usv\_set:nnn {stixfbbit}{L}{E209}  
377 \usv\_set:nnn {stixfbbit}{M}{E20A}  
378 \usv\_set:nnn {stixfbbit}{O}{E20B}  
379 \usv\_set:nnn {stixfbbit}{S}{E20C}  
380 \usv\_set:nnn {stixfbbit}{T}{E20D}  
381 \usv\_set:nnn {stixfbbit}{U}{E20E}  
382 \usv\_set:nnn {stixfbbit}{V}{E20F}  
383 \usv\_set:nnn {stixfbbit}{W}{E210}  
384 \usv\_set:nnn {stixfbbit}{X}{E211}  
385 \usv\_set:nnn {stixfbbit}{Y}{E212}  
  
386 \usv\_set:nnn {stixfbbit}{a}{E213}  
387 \usv\_set:nnn {stixfbbit}{b}{E214}  
388 \usv\_set:nnn {stixfbbit}{c}{E215}  
389 \usv\_set:nnn {stixfbbit}{e}{E217}  
390 \usv\_set:nnn {stixfbbit}{f}{E218}  
391 \usv\_set:nnn {stixfbbit}{g}{E219}  
392 \usv\_set:nnn {stixfbbit}{h}{E21A}  
393 \usv\_set:nnn {stixfbbit}{k}{E21D}  
394 \usv\_set:nnn {stixfbbit}{l}{E21E}  
395 \usv\_set:nnn {stixfbbit}{m}{E21F}  
396 \usv\_set:nnn {stixfbbit}{n}{E220}  
397 \usv\_set:nnn {stixfbbit}{o}{E221}  
398 \usv\_set:nnn {stixfbbit}{p}{E222}  
399 \usv\_set:nnn {stixfbbit}{q}{E223}  
400 \usv\_set:nnn {stixfbbit}{r}{E224}  
401 \usv\_set:nnn {stixfbbit}{s}{E225}  
402 \usv\_set:nnn {stixfbbit}{t}{E226}  
403 \usv\_set:nnn {stixfbbit}{u}{E227}  
404 \usv\_set:nnn {stixfbbit}{v}{E228}  
405 \usv\_set:nnn {stixfbbit}{w}{E229}  
406 \usv\_set:nnn {stixfbbit}{x}{E22A}  
407 \usv\_set:nnn {stixfbbit}{y}{E22B}  
408 \usv\_set:nnn {stixfbbit}{z}{E22C}  
  
409 \usv\_set:nnn {stixbfcalf}{Latin}{E247}  
  
410 \usv\_set:nnn {stixbfitslash}{Latin}{E295}  
411 \usv\_set:nnn {stixbfitslash}{latin}{E2C9}  
412 \usv\_set:nnn {stixbfitslash}{greek}{E32D}  
413 \usv\_set:nnn {stixsfitslash}{epsilon}{E37B}  
414 \usv\_set:nnn {stixsfitslash}{vartheta}{E35F}  
415 \usv\_set:nnn {stixsfitslash}{varkappa}{E375}  
416 \usv\_set:nnn {stixsfitslash}{phi}{E361}  
417 \usv\_set:nnn {stixsfitslash}{varrho}{E377}  
418 \usv\_set:nnn {stixsfitslash}{varpi}{E363}  
419 \usv\_set:nnn {stixsfitslash}{digamma}{E36B}  
  
420 *(/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:nNn` #1 : A L<sup>A</sup>T<sub>E</sub>X 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:nNn
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   \tl_case: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  $\sum$  ( $\Sigma$ ). 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:

$$(\backslash\sum \rightarrow) \Sigma \rightarrow \backslash\sum\_sym \rightarrow \backslash\sumop\nolimits$$

$$(\backslash\int \rightarrow) \int \rightarrow \backslash\int\_sym \rightarrow \backslash\intop$$

```

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} {\updefault}
21  }
22  \SetMathAlphabet #1 {\l_@@_mversion_tl} {\g_fontspec_encoding_tl} {\l_@@_tmpa_tl} {\md-
23  default} {\updefault}
24
25  % integrate with fontspec's \setmathrm etc:
26  \tl_case: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<sub>E</sub>X 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<sub>E</sub>X 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: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.

```

87 \@@_cs_set:Nn \__fontspec_setmathrm_hook:nn
88   {
89     \SetMathAlphabet\mathrm{normal}\g__fontspec_encoding_tl\g__fontspec_mathrm_tl\mddefault\updefa
90     \SetMathAlphabet\mathit{normal}\g__fontspec_encoding_tl\g__fontspec_mathrm_tl\mddefault\itdefa
91     \SetMathAlphabet\mathbf{normal}\g__fontspec_encoding_tl\g__fontspec_mathrm_tl\bfdefault\updefa
92   }
93 \@@_cs_set:Nn \__fontspec_setboldmathrm_hook:nn
94   {
95     \SetMathAlphabet\mathrm{bold}\g__fontspec_encoding_tl\g__fontspec_bfmathrm_tl\mddefault\updefa
96     \SetMathAlphabet\mathbf{bold}\g__fontspec_encoding_tl\g__fontspec_bfmathrm_tl\bfdefault\updefa
97     \SetMathAlphabet\mathit{bold}\g__fontspec_encoding_tl\g__fontspec_bfmathrm_tl\mddefault\itdefa
98   }
99 \@@_cs_set:Nn \__fontspec_setmathsf_hook:nn
100  {
101    \SetMathAlphabet\mathsf{normal}\g__fontspec_encoding_tl\g__fontspec_mathsf_tl\mddefault\updefa
102    \SetMathAlphabet\mathsf{bold} \g__fontspec_encoding_tl\g__fontspec_mathsf_tl\bfdefault\updefa
103  }
104 \@@_cs_set:Nn \__fontspec_setmathtt_hook:nn
105  {
106    \SetMathAlphabet\mathtt{normal}\g__fontspec_encoding_tl\g__fontspec_mathtt_tl\mddefault\updefa
107    \SetMathAlphabet\mathtt{bold} \g__fontspec_encoding_tl\g__fontspec_mathtt_tl\bfdefault\updefa
108  }

```

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

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

```

109 \tl_if_eq:onF {\g__fontspec_mathrm_tl} {\rmdefault} { \__fontspec_setmathrm_hook:nn {} {} }
110 \tl_if_eq:onF {\g__fontspec_mathsf_tl} {\sfdefault} { \__fontspec_setmathsf_hook:nn {} {} }
111 \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)

```

112 \AtBeginDocument
113   {

```



```
114 \tl_if_eq:onT {\g__fontspec_mathrm_tl} {\rmdefault} { \__fontspec_setmathrm_hook:nn {} {} }
115 \tl_if_eq:onT {\g__fontspec_mathsf_tl} {\sfdefault} { \__fontspec_setmathsf_hook:nn {} {} }
116 \tl_if_eq:onT {\g__fontspec_mathtt_tl} {\ttdefault} { \__fontspec_setmathtt_hook:nn {} {} }
117 }
118 </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@currsiz \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) FontAdjustment = { \@@_luatex_copy_fontdimens: },
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>   FontAdjustment = { \@@_luatex_copy_fontdimens: },
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 }

```

`\@@_luatex_copy_fontdimens:` This performs a once-off copy of the LuaTeX math params into XeTeX-like fontdimens. While the list is somewhat comprehensive, these are really only for backwards compatibility and to allow a little shared code. They shouldn't be relied upon, since LuaTeX users might change the math params, which wouldn't be reflected in the fontdimens.

```

153 <*LU>
154 \@@_cs_new:Nn \@@_luatex_copy_fontdimens:
155   {
156     \@@_fontdimen_from_param:nn {10} {ScriptPercentScaleDown}
157     \@@_fontdimen_from_param:nn {11} {ScriptScriptPercentScaleDown}
158     \@@_fontdimen_from_param:nn {15} {AxisHeight}
159     \@@_fontdimen_from_param:nn {18} {SubscriptShiftDown}
160     \@@_fontdimen_from_param:nn {20} {SubscriptBaselineDropMin}
161     \@@_fontdimen_from_param:nn {21} {SuperscriptShiftUp}
162     \@@_fontdimen_from_param:nn {22} {SuperscriptShiftUpCramped}
163     \@@_fontdimen_from_param:nn {24} {SuperscriptBaselineDropMax}
164     \@@_fontdimen_from_param:nn {28} {UpperLimitGapMin}
165     \@@_fontdimen_from_param:nn {29} {UpperLimitBaselineRiseMin}
166     \@@_fontdimen_from_param:nn {30} {LowerLimitGapMin}
167     \@@_fontdimen_from_param:nn {31} {LowerLimitBaselineDropMin}
168     \@@_fontdimen_from_param:nn {32} {StackTopShiftUp}
169     \@@_fontdimen_from_param:nn {42} {FractionNumeratorShiftUp}
170     \@@_fontdimen_from_param:nn {43} {FractionNumeratorDisplayStyleShiftUp}
171     \@@_fontdimen_from_param:nn {44} {FractionDenominatorShiftDown}
172     \@@_fontdimen_from_param:nn {45} {FractionDenominatorDisplayStyleShift-
      Down}
173     \@@_fontdimen_from_param:nn {48} {FractionRuleThickness}
174   }
175 </LU>

```

`\@@_setup_math_fam:`

```

176 \@@_cs_new:Nn \@@_setup_math_fam:
177   {
178     \cs_if_exist:cF { sym \l_@@_symfont_label_tl }
179     {
180       \DeclareSymbolFont{\l_@@_symfont_label_tl}
181         {\encodingdefault}{\l_@@_family_tl}{\mddefault}{\updefault}
182     }
183     \SetSymbolFont{\l_@@_symfont_label_tl}{\l_@@_mversion_tl}
184       {\encodingdefault}{\l_@@_family_tl}{\mddefault}{\updefault}

```

Set the bold math version.

```

185     \str_if_eq:eeT {\l_@@_mversion_tl} {normal}
186     {
187       \SetSymbolFont{\l_@@_symfont_label_tl}{bold}
188         {\encodingdefault}{\l_@@_family_tl}{\bfdefault}{\updefault}
189     }
190   }

```

\@@\_setup\_legacy\_fam\_two: T<sub>E</sub>X 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.

```

191 \@@_cs_new:Nn \@@_setup_legacy_fam_two:
192   {
193     \fontspec_set_family:Nxn \l_@@_fam_two_tl
194     {
195       \l_@@_font_keyval_tl,
196       ScaleAgain = 1.0001,
197       FontAdjustment =
198         {
199           \@@_copy_fontdimen:nnN { 8} {43} \g_@@_main_font_cmd_tl
200           \@@_copy_fontdimen:nnN { 9} {42} \g_@@_main_font_cmd_tl
201           \@@_copy_fontdimen:nnN {10} {32} \g_@@_main_font_cmd_tl
202           \@@_copy_fontdimen:nnN {11} {45} \g_@@_main_font_cmd_tl
203           \@@_copy_fontdimen:nnN {12} {44} \g_@@_main_font_cmd_tl
204           \@@_copy_fontdimen:nnN {13} {21} \g_@@_main_font_cmd_tl
205           \@@_copy_fontdimen:nnN {14} {21} \g_@@_main_font_cmd_tl
206           \@@_copy_fontdimen:nnN {15} {22} \g_@@_main_font_cmd_tl
207           \@@_copy_fontdimen:nnN {16} {18} \g_@@_main_font_cmd_tl
208           \@@_copy_fontdimen:nnN {17} {18} \g_@@_main_font_cmd_tl
209           \@@_copy_fontdimen:nnN {18} {24} \g_@@_main_font_cmd_tl
210           \@@_copy_fontdimen:nnN {19} {20} \g_@@_main_font_cmd_tl
211           \@@_copy_fontdimen:nnN {22} {15} \g_@@_main_font_cmd_tl
212           \@@_zero_fontdimen:n {20} % delim1 = FractionDelimiterDisplaySize
213           \@@_zero_fontdimen:n {21} % delim2 = FractionDelimiterSize
214         }
215     } {\l_@@_fontname_tl}
216
217     \SetSymbolFont{symbols}{\l_@@_mversion_tl}
218     {\encodingdefault}{\l_@@_fam_two_tl}{\mddefault}{\updefault}
219
220     \str_if_eq:eeT {\l_@@_mversion_tl} {normal}
221     {
222       \SetSymbolFont{symbols}{bold}
223       {\encodingdefault}{\l_@@_fam_two_tl}{\bfdefault}{\updefault}
224     }
225   }

```

\@@\_setup\_legacy\_fam\_three: Similarly, this font is shrunk by an imperceptable amount for T<sub>E</sub>X to load it again.

```

226 \@@_cs_new:Nn \@@_setup_legacy_fam_three:
227   {
228     \fontspec_set_family:Nxn \l_@@_fam_three_tl
229     {
230       \l_@@_font_keyval_tl,
231       ScaleAgain = 0.9999,
232       FontAdjustment = {
233         \@@_copy_fontdimen:nnN { 8} {48} \g_@@_main_font_cmd_tl

```

```

234     \@@_copy_fontdimen:nnN { 9} {28} \g_@@_main_font_cmd_tl
235     \@@_copy_fontdimen:nnN {10} {30} \g_@@_main_font_cmd_tl
236     \@@_copy_fontdimen:nnN {11} {29} \g_@@_main_font_cmd_tl
237     \@@_copy_fontdimen:nnN {12} {31} \g_@@_main_font_cmd_tl
238     \@@_zero_fontdimen:n {13}
239   }
240   } {\l_@@_fontname_tl}
241
242   \SetSymbolFont{largesymbols}{\l_@@_mversion_tl}
243     {\encodingdefault}{\l_@@_fam_three_tl}{\mddefault}{\updefault}
244
245   \str_if_eq:eeT {\l_@@_mversion_tl} {normal}
246   {
247     \SetSymbolFont{largesymbols}{bold}
248       {\encodingdefault}{\l_@@_fam_three_tl}{\bfdefault}{\updefault}
249   }
250 }

```

\@@\_onceoff\_setup:

```

251 \@@_cs_new:Nn \@@_onceoff_setup:
252 {
253   \@@_set_delcode:nnn {operators} {\`.} {0}
254 }

```

## 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.

\@@\_process\_symbol\_parse:nnn

```

255 \cs_set:Nn \@@_process_symbol_noparse:nnn
256 {
257   \@@_set_mathsymbol:nNn {\l_@@_symfont_label_tl} #2 #3 {#1}
258 }
259 \cs_set:Nn \@@_process_symbol_parse:nnn
260 {
261   \@@_if_char_spec:nNT {#1} {#3}
262   {
263     \@@_process_symbol_noparse:nnn {#1} {#2} {#3}
264   }
265 }

```

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

```

266 \@@_cs_new:Nn \@@_remap_symbols:
267 {
268   \@@_remap_symbol:nnn {\-} {\mathbin} {"2212}
269   \@@_remap_symbol:nnn {\*} {\mathbin} {"02217}% text asterisk to "cen-
    tred asterisk"
270   \bool_if:NF \g_@@_literal_colon_bool

```



```

271     {
272         \@@_remap_symbol:nnn {\:} {\mathrel} {"02236}% colon to ra-
        tio (i.e., punct to rel)
273     }
274 }

```

\@@\_remap\_symbol\_noparse:nnn Where \@@\_remap\_symbol:nnn is defined to be one of these two, depending on the  
\@@\_remap\_symbol\_parse:nnn range setup:

```

275 \cs_new:Nn \@@_remap_symbol_parse:nnn
276 {
277     \@@_if_char_spec:nNT {#3} {#2}
278     { \@@_remap_symbol_noparse:nnn {#1} {#2} {#3} }
279 }
280 \cs_new:Nn \@@_remap_symbol_noparse:nnn
281 {
282     \clist_map_inline:nn {#1}
283     { \@@_set_mathcode:nnnn {##1} {#2} {\l_@@_symfont_label_tl} {#3} }
284 }

```

## 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.

```

285 \@@_cs_new:Nn \@@_setup_mathactives:
286 {
287     \@@_make_mathactive:nNN {"2032} \@@_prime_single_mchar \mathord
288     \@@_make_mathactive:nNN {"2033} \@@_prime_double_mchar \mathord
289     \@@_make_mathactive:nNN {"2034} \@@_prime_triple_mchar \mathord
290     \@@_make_mathactive:nNN {"2057} \@@_prime_quad_mchar \mathord
291     \@@_make_mathactive:nNN {"2035} \@@_backprime_single_mchar \mathord
292     \@@_make_mathactive:nNN {"2036} \@@_backprime_double_mchar \mathord
293     \@@_make_mathactive:nNN {"2037} \@@_backprime_triple_mchar \mathord
294     \@@_make_mathactive:nNN {\`'} \mathstraightquote \mathord
295     \@@_make_mathactive:nNN {\`}`} \mathbacktick \mathord
296 }

```

\@@\_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!

```

297 \cs_new:Nn \@@_make_mathactive_parse:nNN
298 {
299     \@@_if_char_spec:nNT {#1} #3
300     { \@@_make_mathactive_noparse:nNN {#1} #2 #3 }
301 }
302 \cs_new:Nn \@@_make_mathactive_noparse:nNN
303 {

```

```

304 \@@_set_mathchar:NNnn #2 #3 {\l_@@_symfont_label_tl} {#1}
305 \@@_char_gmake_mathactive:n {#1}
306 }

```

### 12.3 Delimiter codes

\@@\_assign\_delcode:nn

```

307 \cs_new:Nn \@@_assign_delcode_noparse:nn
308 {
309   \@@_set_delcode:nnn \l_@@_symfont_label_tl {#1} {#2}
310 }
311 \cs_new:Nn \@@_assign_delcode_parse:nn
312 {
313   \@@_if_char_spec:nNT {#2} \@nil
314   {
315     \@@_assign_delcode_noparse:nn {#1} {#2}
316   }
317 }

```

\@@\_assign\_delcode:n Shorthand.

```

318 \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.

```

319 \@@_cs_new:Nn \@@_setup_delcodes:
320 {
321   \@@_assign_delcode:nn {'\'} {\g_@@_slash_delimiter_usv}
322   \@@_assign_delcode:nn {"2044} {\g_@@_slash_delimiter_usv} % fracslash
323   \@@_assign_delcode:nn {"2215} {\g_@@_slash_delimiter_usv} % divslash
324   \@@_assign_delcode:n {"005C} % backslash
325   \@@_assign_delcode:nn {'\<} {"27E8} % angle brackets with ascii notation
326   \@@_assign_delcode:nn {'\>} {"27E9} % angle brackets with ascii notation
327   \@@_assign_delcode:n {"2191} % up arrow
328   \@@_assign_delcode:n {"2193} % down arrow
329   \@@_assign_delcode:n {"2195} % updown arrow
330   \@@_assign_delcode:n {"219F} % up arrow twohead
331   \@@_assign_delcode:n {"21A1} % down arrow twohead
332   \@@_assign_delcode:n {"21A5} % up arrow from bar
333   \@@_assign_delcode:n {"21A7} % down arrow from bar
334   \@@_assign_delcode:n {"21A8} % updown arrow from bar
335   \@@_assign_delcode:n {"21BE} % up harpoon right
336   \@@_assign_delcode:n {"21BF} % up harpoon left
337   \@@_assign_delcode:n {"21C2} % down harpoon right
338   \@@_assign_delcode:n {"21C3} % down harpoon left
339   \@@_assign_delcode:n {"21C5} % arrows up down
340   \@@_assign_delcode:n {"21F5} % arrows down up

```

```

341 \@@_assign_delcode:n {"21C8} % arrows up up
342 \@@_assign_delcode:n {"21CA} % arrows down down
343 \@@_assign_delcode:n {"21D1} % double up arrow
344 \@@_assign_delcode:n {"21D3} % double down arrow
345 \@@_assign_delcode:n {"21D5} % double updown arrow
346 \@@_assign_delcode:n {"21DE} % up arrow double stroke
347 \@@_assign_delcode:n {"21DF} % down arrow double stroke
348 \@@_assign_delcode:n {"21E1} % up arrow dashed
349 \@@_assign_delcode:n {"21E3} % down arrow dashed
350 \@@_assign_delcode:n {"21E7} % up white arrow
351 \@@_assign_delcode:n {"21E9} % down white arrow
352 \@@_assign_delcode:n {"21EA} % up white arrow from bar
353 \@@_assign_delcode:n {"21F3} % updown white arrow
354 }

```

## 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.

```

355 \tl_set:Nn \l_@@_nolimits_tl
356 {
357   \int\iint\iiint\iiiint\oint\oiint\oiiint
358   \intclockwise\varointclockwise\ointctrlockwise\sumint
359   \intbar\intBar\find\cirfnint\awint\rppoint
360   \scpolint\mpoint\pointint\sqint\intlarhk\intx
361   \intcap\intcup\upint\lowint
362 }

```

## 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.)

```

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

```

## 12.6 Fontdimens

```

364 (*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.

```

365 \@@_cs_new:Nn \@@_mathparam_restore:
366 {
367   \glb@settings
368   \tl_use:N \g_@@_mathparam_settings_tl
369 }

```

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

```

370 \@@_cs_new:Nn \@@_mathparam_store:
371 {
372   \glb@settings
373   \tl_gset:Nx \g_@@_mathparam_settings_tl
374   {
375     \@@_mathparam_store_aux:N \displaystyle
376     \@@_mathparam_store_aux:N \textstyle
377     \@@_mathparam_store_aux:N \scriptstyle
378     \@@_mathparam_store_aux:N \scriptscriptstyle
379   }
380 }

381 \cs_set:Nn \@@_mathparam_store_aux:N
382 {
383   \Umathquad          #1 = \the \Umathquad          #1 \scan_stop:
384   \Umathaxis          #1 = \the \Umathaxis          #1 \scan_stop:
385   \Umathoperatorsize #1 = \the \Umathoperatorsize #1 \scan_stop:
386   \Umathoverbarkern  #1 = \the \Umathoverbarkern  #1 \scan_stop:
387   \Umathoverbarrule  #1 = \the \Umathoverbarrule  #1 \scan_stop:
388   \Umathoverbarvgap  #1 = \the \Umathoverbarvgap  #1 \scan_stop:
389   \Umathunderbarkern #1 = \the \Umathunderbarkern #1 \scan_stop:
390   \Umathunderbarrule #1 = \the \Umathunderbarrule #1 \scan_stop:
391   \Umathunderbarvgap #1 = \the \Umathunderbarvgap #1 \scan_stop:
392   \Umathradicalkern  #1 = \the \Umathradicalkern  #1 \scan_stop:
393   \Umathradicalrule  #1 = \the \Umathradicalrule  #1 \scan_stop:
394   \Umathradicalvgap  #1 = \the \Umathradicalvgap  #1 \scan_stop:
395   \Umathradicaldegreebefore #1 = \the \Umathradicaldegreebefore #1 \scan_stop:
396   \Umathradicaldegreeafter #1 = \the \Umathradicaldegreeafter #1 \scan_stop:
397   \Umathradicaldegreerise #1 = \the \Umathradicaldegreerise #1 \scan_stop:
398   \Umathstackvgap    #1 = \the \Umathstackvgap    #1 \scan_stop:
399   \Umathstacknumup   #1 = \the \Umathstacknumup   #1 \scan_stop:
400   \Umathstackdenomdown #1 = \the \Umathstackdenomdown #1 \scan_stop:
401   \Umathfractionrule #1 = \the \Umathfractionrule #1 \scan_stop:
402   \Umathfractionnumvgap #1 = \the \Umathfractionnumvgap #1 \scan_stop:
403   \Umathfractionnumup #1 = \the \Umathfractionnumup #1 \scan_stop:
404   \Umathfractiondenomvgap #1 = \the \Umathfractiondenomvgap #1 \scan_stop:
405   \Umathfractiondenomdown #1 = \the \Umathfractiondenomdown #1 \scan_stop:
406   \Umathfractiondelsize #1 = \the \Umathfractiondelsize #1 \scan_stop:

```

```

407 \Umathlimitabovevgap #1 = \the \Umathlimitabovevgap #1 \scan_stop:
408 \Umathlimitabovebgap #1 = \the \Umathlimitabovebgap #1 \scan_stop:
409 \Umathlimitabovekern #1 = \the \Umathlimitabovekern #1 \scan_stop:
410 \Umathlimitbelowvgap #1 = \the \Umathlimitbelowvgap #1 \scan_stop:
411 \Umathlimitbelowbgap #1 = \the \Umathlimitbelowbgap #1 \scan_stop:
412 \Umathlimitbelowkern #1 = \the \Umathlimitbelowkern #1 \scan_stop:
413 \Umathoverdelimitervgap #1 = \the \Umathoverdelimitervgap #1 \scan_stop:
414 \Umathoverdelimiterbgap #1 = \the \Umathoverdelimiterbgap #1 \scan_stop:
415 \Umathunderdelimitervgap #1 = \the \Umathunderdelimitervgap #1 \scan_stop:
416 \Umathunderdelimiterbgap #1 = \the \Umathunderdelimiterbgap #1 \scan_stop:
417 \Umathsubshiftdrop #1 = \the \Umathsubshiftdrop #1 \scan_stop:
418 \Umathsubshiftdown #1 = \the \Umathsubshiftdown #1 \scan_stop:
419 \Umathsupshiftdrop #1 = \the \Umathsupshiftdrop #1 \scan_stop:
420 \Umathsupshiftdown #1 = \the \Umathsupshiftdown #1 \scan_stop:
421 \Umathsubsupshiftdown #1 = \the \Umathsubsupshiftdown #1 \scan_stop:
422 \Umathsubtopmax #1 = \the \Umathsubtopmax #1 \scan_stop:
423 \Umathsupbottommin #1 = \the \Umathsupbottommin #1 \scan_stop:
424 \Umathsupsubbottommax #1 = \the \Umathsupsubbottommax #1 \scan_stop:
425 \Umathsubsupvgap #1 = \the \Umathsubsupvgap #1 \scan_stop:
426 \Umathspaceafterscript #1 = \the \Umathspaceafterscript #1 \scan_stop:
427 \Umathconnectoroverlapmin #1 = \the \Umathconnectoroverlapmin #1 \scan_stop:
428 }

429 </LU>
430 </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     }
```

```

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:NnTF \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:nnn
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<sub>Y</sub>TeX 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<sup>A</sup>TeX3-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  $\@@_\langle\text{L}^{\text{A}}\text{T}^{\text{E}}\text{X}3\text{ name}\rangle:N$  and a protected setter command  $\@@\_set_\langle\text{L}^{\text{A}}\text{T}^{\text{E}}\text{X}3\text{ name}\rangle:Nn$ . 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<sub>Y</sub>TeX doesn't seem to provide  $\font\dimens$ ; 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<sub>Y</sub>TeX, 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<sub>Y</sub>TeX, 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
  \crampedscriptscriptstyle 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 X $\TeX$ , 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 X $\TeX$  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  $\Umath$ . 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  $\Umath$ .

```

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:n { 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:n` takes a font dimension number and outputs the decimal value of the associated parameter. `\@@_fontdimen_pc_to_pt:n` 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:n {10} #4 #3
158   \else
159     \ifx#1\scriptscriptstyle
160       #2 \@@_fontdimen_to_percent:n {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:nn {##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 definition.

```
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} {scr }

```

```

41 \@@_default_mathalph:nnn {bfc al } {Latin}                {bfscr }
42 \@@_default_mathalph:nnn {frac } {latin, Latin}          {frac }
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, frac, 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, frac, 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 caligraphic: 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 }

```

### 17.5 *Fraktur or fraktur or blackletter: frak*

```

238 \@@_new_alphabet_config:nnn {frak} {Latin}
239 {
240 \@@_set_mathalphabet_Latin:nnn {frak} {up,it} {#1}
241 \@@_set_mathalphabet_pos:nnnn {frak} {C} {up,it} {#1}
242 \@@_set_mathalphabet_pos:nnnn {frak} {H} {up,it} {#1}
243 \@@_set_mathalphabet_pos:nnnn {frak} {I} {up,it} {#1}
244 \@@_set_mathalphabet_pos:nnnn {frak} {R} {up,it} {#1}
245 \@@_set_mathalphabet_pos:nnnn {frak} {Z} {up,it} {#1}
246 }
247 \@@_new_alphabet_config:nnn {frak} {latin}
248 {
249 \@@_set_mathalphabet_latin:nnn {frak} {up,it} {#1}
250 }

```

### 17.6 *Sans serif upright: sfup*

```

251 \@@_new_alphabet_config:nnn {sfup} {num}
252 {
253 \@@_set_mathalphabet_numbers:nnn {sf} {up} {#1}
254 \@@_set_mathalphabet_numbers:nnn {sfup} {up} {#1}
255 }
256 \@@_new_alphabet_config:nnn {sfup} {Latin}
257 {
258 \bool_if:NTF \g_@@_sfliteral_bool
259 {
260 \@@_set_normal_Latin:nn {sfup} {#1}
261 \@@_set_mathalphabet_Latin:nnn {sf} {up} {#1}
262 }
263 {
264 \bool_if:NT \g_@@_upsans_bool
265 {
266 \@@_set_normal_Latin:nn {sfup,sfit} {#1}
267 \@@_set_mathalphabet_Latin:nnn {sf} {up,it} {#1}
268 }
269 }
270 \@@_set_mathalphabet_Latin:nnn {sfup} {up,it} {#1}
271 }
272
273 \@@_new_alphabet_config:nnn {sfup} {latin}

```

```

274 {
275   \bool_if:NTF \g_@@_sfliteral_bool
276   {
277     \@@_set_normal_latin:nn {sfup} {#1}
278     \@@_set_mathalphabet_latin:nnn {sf} {up} {#1}
279   }
280   {
281     \bool_if:NT \g_@@_upsans_bool
282     {
283       \@@_set_normal_latin:nn {sfup,sfit} {#1}
284       \@@_set_mathalphabet_latin:nnn {sf} {up,it} {#1}
285     }
286   }
287   \@@_set_mathalphabet_latin:nnn {sfup} {up,it} {#1}
288 }

```

### 17.7 *Sans serif italic: sfit*

```

289 \@@_new_alphabet_config:nnn {sfit} {Latin}
290 {
291   \bool_if:NTF \g_@@_sfliteral_bool
292   {
293     \@@_set_normal_Latin:nn {sfit} {#1}
294     \@@_set_mathalphabet_Latin:nnn {sf} {it} {#1}
295   }
296   {
297     \bool_if:NF \g_@@_upsans_bool
298     {
299       \@@_set_normal_Latin:nn {sfup,sfit} {#1}
300       \@@_set_mathalphabet_Latin:nnn {sf} {up,it} {#1}
301     }
302   }
303   \@@_set_mathalphabet_Latin:nnn {sfit} {up,it} {#1}
304 }
305
306 \@@_new_alphabet_config:nnn {sfit} {latin}
307 {
308   \bool_if:NTF \g_@@_sfliteral_bool
309   {
310     \@@_set_normal_latin:nn {sfit} {#1}
311     \@@_set_mathalphabet_latin:nnn {sf} {it}{#1}
312   }
313   {
314     \bool_if:NF \g_@@_upsans_bool
315     {
316       \@@_set_normal_latin:nn {sfup,sfit} {#1}
317       \@@_set_mathalphabet_latin:nnn {sf} {up,it}{#1}
318     }
319   }
320   \@@_set_mathalphabet_latin:nnn {sfit} {up,it}{#1}

```

321 }

## 17.8 *Typewriter or monospaced: tt*

```
322 \@@_new_alphabet_config:nnn {tt} {num}
323 {
324   \@@_set_mathalphabet_numbers:nnn {tt} {up}{#1}
325 }
326 \@@_new_alphabet_config:nnn {tt} {Latin}
327 {
328   \@@_set_mathalphabet_Latin:nnn {tt} {up,it}{#1}
329 }
330 \@@_new_alphabet_config:nnn {tt} {latin}
331 {
332   \@@_set_mathalphabet_latin:nnn {tt} {up,it}{#1}
333 }
```

## 17.9 *Bold Italic: bfit*

```
334 \@@_new_alphabet_config:nnn {bfit} {Latin}
335 {
336   \bool_if:NF \g_@@_bfupLatin_bool
337   {
338     \@@_set_normal_Latin:nn {bfup,bfit} {#1}
339   }
340   \@@_set_mathalphabet_Latin:nnn {bfit} {up,it}{#1}
341   \bool_if:NTF \g_@@_bfliteral_bool
342   {
343     \@@_set_normal_Latin:nn {bfit} {#1}
344     \@@_set_mathalphabet_Latin:nnn {bf} {it}{#1}
345   }
346   {
347     \bool_if:NF \g_@@_bfupLatin_bool
348     {
349       \@@_set_normal_Latin:nn {bfup,bfit} {#1}
350       \@@_set_mathalphabet_Latin:nnn {bf} {up,it}{#1}
351     }
352   }
353 }
354
355 \@@_new_alphabet_config:nnn {bfit} {latin}
356 {
357   \bool_if:NF \g_@@_bfuplatin_bool
358   {
359     \@@_set_normal_latin:nn {bfup,bfit} {#1}
360   }
361   \@@_set_mathalphabet_latin:nnn {bfit} {up,it}{#1}
362   \bool_if:NTF \g_@@_bfliteral_bool
363   {
364     \@@_set_normal_latin:nn {bfit} {#1}
365     \@@_set_mathalphabet_latin:nnn {bf} {it}{#1}
366   }
367 }
```

```

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

```



```

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

```

### 17.10 Bold Upright: *bfup*

```

451 \@@_new_alphabet_config:nnn {bfup} {num}
452 {
453   \@@_set_mathalphabet_numbers:nnn {bf} {up} {#1}
454   \@@_set_mathalphabet_numbers:nnn {bfup} {up} {#1}
455 }
456
457 \@@_new_alphabet_config:nnn {bfup} {Latin}
458 {
459   \bool_if:NT \g_@@_bfupLatin_bool
460   {
461     \@@_set_normal_Latin:nn {bfup,bfit} {#1}

```

```

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

```

```

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

```

```

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

```

### 17.11 *Bold fractur or fraktur or blackletter: bffrak*

```

582 \@@_new_alphabet_config:nnn {bffrak} {Latin}
583 {
584   \@@_set_mathalphabet_Latin:nnn {bffrak} {up,it}{#1}
585 }
586
587 \@@_new_alphabet_config:nnn {bffrak} {latin}
588 {
589   \@@_set_mathalphabet_latin:nnn {bffrak} {up,it}{#1}
590 }

```

### 17.12 *Bold script or calligraphic: bfscr*

```

591 \@@_new_alphabet_config:nnn {bfscr} {Latin}
592 {
593   \@@_set_mathalphabet_Latin:nnn {bfscr} {up,it}{#1}
594 }
595 \@@_new_alphabet_config:nnn {bfscr} {latin}
596 {
597   \@@_set_mathalphabet_latin:nnn {bfscr} {up,it}{#1}
598 }
599 \@@_new_alphabet_config:nnn {bfcal} {Latin}
600 {
601   \@@_set_mathalphabet_Latin:nnn {bfcal} {up,it}{#1}
602 }

```

### 17.13 *Bold upright sans serif: bfsfup*

```

603 \@@_new_alphabet_config:nnn {bfsfup} {num}
604 {
605   \@@_set_mathalphabet_numbers:nnn {bfsf} {up}{#1}
606   \@@_set_mathalphabet_numbers:nnn {bfsfup} {up}{#1}
607 }
608 \@@_new_alphabet_config:nnn {bfsfup} {Latin}
609 {
610   \bool_if:NTF \g_@@_sfliteral_bool
611   {
612     \@@_set_normal_Latin:nn {bfsfup} {#1}
613     \@@_set_mathalphabet_Latin:nnn {bfsf} {up}{#1}
614   }
615   {
616     \bool_if:NT \g_@@_upsans_bool
617     {
618       \@@_set_normal_Latin:nn {bfsfup,bfsfit} {#1}
619       \@@_set_mathalphabet_Latin:nnn {bfsf} {up,it}{#1}
620     }
621   }
622   \@@_set_mathalphabet_Latin:nnn {bfsfup} {up,it}{#1}
623 }
624
625 \@@_new_alphabet_config:nnn {bfsfup} {latin}
626 {
627   \bool_if:NTF \g_@@_sfliteral_bool
628   {
629     \@@_set_normal_latin:nn {bfsfup} {#1}
630     \@@_set_mathalphabet_latin:nnn {bfsf} {up}{#1}
631   }
632   {
633     \bool_if:NT \g_@@_upsans_bool
634     {
635       \@@_set_normal_latin:nn {bfsfup,bfsfit} {#1}
636       \@@_set_mathalphabet_latin:nnn {bfsf} {up,it}{#1}
637     }
638   }
639   \@@_set_mathalphabet_latin:nnn {bfsfup} {up,it}{#1}
640 }
641
642 \@@_new_alphabet_config:nnn {bfsfup} {Greek}
643 {
644   \bool_if:NTF \g_@@_sfliteral_bool
645   {
646     \@@_set_normal_Greek:nn {bfsfup}{#1}
647     \@@_set_mathalphabet_Greek:nnn {bfsf} {up}{#1}
648   }
649   {
650     \bool_if:NT \g_@@_upsans_bool
651     {

```

```

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

```

```

701 {
702   \@@_set_mathalphabet_pos:nnnn {bfsf} {partial} {up}{#1}
703 }
704 {
705   \bool_if:NT \g_@@_uppartial_bool
706   {
707     \@@_set_mathalphabet_pos:nnnn {bfsf} {partial} {up,it}{#1}
708   }
709 }
710 \bool_if:NTF \g_@@_literal_Nabla_bool
711 {
712   \@@_set_mathalphabet_pos:nnnn {bfsf} {Nabla} {up}{#1}
713 }
714 {
715   \bool_if:NT \g_@@_upNabla_bool
716   {
717     \@@_set_mathalphabet_pos:nnnn {bfsf} {Nabla} {up,it}{#1}
718   }
719 }
720 }

```

#### 17.14 *Bold italic sans serif: bfsfit*

```

721 \@@_new_alphabet_config:nnn {bfsfit} {Latin}
722 {
723   \bool_if:NTF \g_@@_sfliteral_bool
724   {
725     \@@_set_normal_Latin:nn {bfsfit} {#1}
726     \@@_set_mathalphabet_Latin:nnn {bfsf} {it}{#1}
727   }
728   {
729     \bool_if:NF \g_@@_upsans_bool
730     {
731       \@@_set_normal_Latin:nn {bfsfup,bfsfit} {#1}
732       \@@_set_mathalphabet_Latin:nnn {bfsf} {up,it}{#1}
733     }
734   }
735   \@@_set_mathalphabet_Latin:nnn {bfsfit} {up,it}{#1}
736 }
737
738 \@@_new_alphabet_config:nnn {bfsfit} {latin}
739 {
740   \bool_if:NTF \g_@@_sfliteral_bool
741   {
742     \@@_set_normal_latin:nn {bfsfit} {#1}
743     \@@_set_mathalphabet_latin:nnn {bfsf} {it}{#1}
744   }
745   {
746     \bool_if:NF \g_@@_upsans_bool
747     {

```

```

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

```



```

797     {
798       \@_set_normal_char:nnn {Nabla}{bfsfup,bfsfit}{#1}
799     }
800   }
801   \bool_if:NTF \g_@@_literal_partial_bool
802   {
803     \@_set_normal_char:nnn {partial}{bfsfit}{#1}
804   }
805   {
806     \bool_if:NF \g_@@_uppartial_bool
807     {
808       \@_set_normal_char:nnn {partial}{bfsfup,bfsfit}{#1}
809     }
810   }
811   \@_set_mathalphabet_pos:nnnn {bfsfit} {partial} {up,it}{#1}
812   \@_set_mathalphabet_pos:nnnn {bfsfit} {Nabla} {up,it}{#1}
813   \bool_if:NTF \g_@@_literal_partial_bool
814   {
815     \@_set_mathalphabet_pos:nnnn {bfsf} {partial} {it}{#1}
816   }
817   {
818     \bool_if:NF \g_@@_uppartial_bool
819     {
820       \@_set_mathalphabet_pos:nnnn {bfsf} {partial} {up,it}{#1}
821     }
822   }
823   \bool_if:NTF \g_@@_literal_Nabla_bool
824   {
825     \@_set_mathalphabet_pos:nnnn {bfsf} {Nabla} {it}{#1}
826   }
827   {
828     \bool_if:NF \g_@@_upNabla_bool
829     {
830       \@_set_mathalphabet_pos:nnnn {bfsf} {Nabla} {up,it}{#1}
831     }
832   }
833 }
834 </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  $\text{\LaTeX}$ , 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 \
261   \char_set_catcode_active:N \
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 '      \@@_scan_sup_prime:
272       \cs_set_eq:NN ^^^^2032 \@@_scan_sup_prime:
273       \cs_set_eq:NN ^^^^2033 \@@_scan_sup_dprime:
274       \cs_set_eq:NN ^^^^2034 \@@_scan_sup_trprime:
275       \cs_set_eq:NN ^^^^2057 \@@_scan_sup_qprime:
276       \cs_set_eq:NN `     \@@_scan_sup_backprime:
277       \cs_set_eq:NN ^^^^2035 \@@_scan_sup_backprime:
278       \cs_set_eq:NN ^^^^2036 \@@_scan_sup_backdprime:
279       \cs_set_eq:NN ^^^^2037 \@@_scan_sup_backtrprime:
280     }
281 \group_end:
282 </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)
```

```
\@@_check_and_fix:NNnnn #1 : command  
#2 : factory command  
#3 : parameter text  
#4 : expected replacement text  
#5 : new replacement text
```

Tries to patch  $\langle command \rangle$ . If  $\langle command \rangle$  is undefined, do nothing. Otherwise it must be a macro with the given  $\langle parameter text \rangle$  and  $\langle expected replacement text \rangle$ , created by the given  $\langle factory command \rangle$  or equivalent. In this case it will be overwritten using the  $\langle parameter text \rangle$  and the  $\langle new replacement text for LuaTeX \rangle$  or the  $\langle new replacement text for XeTeX \rangle$ , depending on the engine. Otherwise issue a warning and don't overwrite.

```
2 \cs_new_protected_nopar:Nn \@@_check_and_fix:NNnnn  
3 {  
4   \cs_if_exist:NT #1  
5   {  
6     \token_if_macro:NTF #1  
7     {  
8       \group_begin:  
9       #2 \@@_tmpa:w #3 { #4 }  
10      \cs_if_eq:NNTF #1 \@@_tmpa:w  
11      {  
12        \msg_info:nxx { unicode-math } { patch-macro } { \to-  
ken_to_str:N #1 }  
13        \group_end:  
14        #2 #1 #3 { #5 }  
15      }  
16      {  
17        \msg_warning:nxxx { unicode-math } { wrong-meaning }  
18        { \token_to_str:N #1 } { \token_to_meaning:N #1 }  
19        { \token_to_meaning:N \@@_tmpa:w }  
20        \group_end:  
21      }  
22    }  
23    {  
24      \msg_warning:nxx { unicode-math } { macro-expected }  
25      { \token_to_str:N #1 }  
26    }  
27  }  
28 }
```

## 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.

```
29 \AtEndOfPackageFile * {url}
30   {
31     \tl_put_left:Nn \Url@FormatString { \@@_switch_to:n {literal} }
32     \tl_put_right:Nn \UrlSpecials
33       {
34         \do \` { \mathchar`` }
35         \do \' { \mathchar``' }
36         \do \$ { \mathchar``\$ }
37         \do \& { \mathchar``& }
38       }
39   }
```

### 21.2 *mathtools*

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

The XeTeX 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.

```
40 (*XE)
41 \AtEndOfPackageFile * { mathtools }
42   {
43     \@@_check_and_fix:NNnnn \MT_cramped_internal:Nn \cs_set_nopar:Npn { #1 #2 }
44     {
45       \sbox \z@ { $ \m@th #1 \nulldelimiterspace = \z@ \radical \z@ { #2 } $ }
46       \ifx #1 \displaystyle
47         \dimen@ = \fontdimen 8 \textfont 3
48         \advance \dimen@ .25 \fontdimen 5 \textfont 2
49       \else
50         \dimen@ = 1.25 \fontdimen 8
51         \ifx #1 \textstyle
52           \textfont
53         \else
54           \ifx #1 \scriptstyle
55             \scriptfont
56           \else
57             \scriptscriptfont
```

```

58     \fi
59     \fi
60     3
61     \fi
62     \advance \dimen@ -\ht\z@
63     \ht\z@ = -\dimen@
64     \box\z@
65   }
66   {
67     \hbox_set:Nn \l_tmpa_box
68     {
69       \color@setgroup \c_math_toggle_token \m@th
70       #1
71       \dim_zero:N \nulldelimiterspace
72       \XeTeXradical 255 ~ 0 ~ { #2 }
73       \c_math_toggle_token \color@endgroup
74     }
75     \box_set_ht:Nn \l_tmpa_box
76     {
77       \box_ht:N \l_tmpa_box - \@_radical_vgap:N #1
78     }
79     \box_use_drop:N \l_tmpa_box
80   }
81 }
82 </XE>

```

`\overbracket` and `\underbracket` are defined in terms of rules, so we keep them, and rename ours to `\Uoverbracket` and `\Uunderbracket`.

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.

```

83 \AtEndOfPackageFile * { mathtools }
84 {
85   \cs_set_eq:NN \MTOverbracket \overbracket
86   \cs_set_eq:NN \MTUnderbracket \underbracket
87
88   \AtBeginDocument
89   {
90     \msg_warning:nn { unicode-math } { mathtools-overbracket }
91
92     \cs_set:Npn \downbracketfill #1 #2
93     {
94       \tl_set:Nn \l_MT_bracketheight_fdim {.27ex}
95       \downbracketend {#1} {#2}
96       \leaders \vrule \@height #1 \@depth \z@ \hfill
97       \downbracketend {#1} {#2}
98     }
99

```

```

100     \cs_set:Npn \upbracketfill #1 #2
101     {
102         \tl_set:Nn \l_MT_bracketheight_fdim {.27ex}
103         \upbracketend {#1} {#2}
104         \leaders \vrule \@height \z@ \@depth #1 \hfill
105         \upbracketend {#1} {#2}
106     }
107
108     \cs_set_eq:NN \Uoverbracket \overbracket
109     \cs_set_eq:NN \Underbracket \underbracket
110     \cs_set_eq:NN \overbracket \MToverbracket
111     \cs_set_eq:NN \underbracket \MTunderbracket
112 }
113 }

```

`\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` Because `mathtools` uses `\providecommand` `\AtBeginDocument`, we can just define `\eqqcolon` the offending commands here.

```

114 \AtEndOfPackageFile * { mathtools }
115 {
116     \msg_warning:nn { unicode-math } { mathtools-colon }
117     \NewDocumentCommand \dblcolon { } { \Colon }
118     \NewDocumentCommand \coloneqq { } { \coloneq }
119     \NewDocumentCommand \Coloneqq { } { \Coloneq }
120     \NewDocumentCommand \eqqcolon { } { \eqcolon }
121 }

```

### 21.3 *colonequals*

`\ratio` Similarly to `mathtools`, the `colonequals` defines several colon combinations. Fortunately there are no name clashes, so we can just overwrite their definitions.

```

\coloncolon 122 \AtEndOfPackageFile * { colonequals }
\colonequals 123 {
\equalscolon 124     \msg_warning:nn { unicode-math } { colonequals }
\coloncolonequals 125     \RenewDocumentCommand \ratio { } { \mathratio }
126     \RenewDocumentCommand \coloncolon { } { \Colon }
127     \RenewDocumentCommand \minuscolon { } { \dashcolon }
128     \RenewDocumentCommand \colonequals { } { \coloneq }
129     \RenewDocumentCommand \equalscolon { } { \eqcolon }
130     \RenewDocumentCommand \coloncolonequals { } { \Coloneq }
131 }
132 (</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 \AtEndOfPackageFile * {amsmath}
3 {
4   \tl_remove_once:Nn \@begindocumenthook
5   {
6     \mathchardef\std@minus\mathcode`\-\relax
7     \mathchardef\std@equal\mathcode`\=\relax
8   }
9   \AtBeginDocument
10  {
11    \Umathcharnumdef\std@minus\Umathcodenum`\-
12    \Umathcharnumdef\std@equal\Umathcodenum`=
13  }
14  \cs_set:Npn \@cdots {\mathinner{\unicodecdots}}
15  \cs_set_eq:NN \dotsb@ \cdots
```

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

```
16 (*XE)
17 \def \resetMathstrut@
18   {%
19   \setbox\z@\hbox{${$}%}
20   \ht\Mathstrutbox@ \ht\z@ \dp\Mathstrutbox@ \dp\z@
21   }
```

The subarray environment uses inappropriate font dimensions.

```
22 \@_check_and_fix:NNnn \subarray \cs_set:Npn { #1 }
23 {
24   \vcenter
25   \bgroup
26   \Let@
27   \restore@math@cr
28   \default@tag
29   \baselineskip \fontdimen 10~ \scriptfont \tw@
30   \advance \baselineskip \fontdimen 12~ \scriptfont \tw@
31   \lineskip \thr@@@ \fontdimen 8~ \scriptfont \thr@@@
32   \lineskiplimit \lineskip
33   \ialign
34   \bgroup
35   \ifx c #1 \hfil \fi
36   $ \m@th \scriptstyle ## $
37   \hfil
```

```

38     \crrc
39   }
40   {
41     \vcenter
42     \c_group_begin_token
43     \Let@
44     \restore@math@cr
45     \default@tag
46     \skip_set:Nn \baselineskip
47     {

```

Here we use stack top shift + stack bottom shift, which sounds reasonable.

```

48       \@@_stack_num_up:N \scriptstyle
49       + \@@_stack_denom_down:N \scriptstyle
50     }

```

Here we use the minimum stack gap.

```

51     \lineskip \@@_stack_vgap:N \scriptstyle
52     \lineskiplimit \lineskip
53     \ialign
54     \c_group_begin_token
55     \token_if_eq_meaning:NNT c #1 { \hfil }
56     \c_math_toggle_token
57     \m@th
58     \scriptstyle
59     \c_parameter_token \c_parameter_token
60     \c_math_toggle_token
61     \hfil
62     \crrc
63   }
64 </XE>

```

The roots need a complete rework.

```

65 < *LU >
66 \@@_check_and_fix:NNnn \plainroot@ \cs_set_nopar:Npn { #1 \of #2 }
67   {
68     \setbox \rootbox \hbox
69     {
70       $ \m@th \scriptscriptstyle { #1 } $
71     }
72     \mathchoice
73     { \r@@@t \displaystyle { #2 } }
74     { \r@@@t \textstyle { #2 } }~
75     { \r@@@t \scriptstyle { #2 } }
76     { \r@@@t \scriptscriptstyle { #2 } }
77     \egroup
78   }
79   {
80     \bool_if:nTF
81     {
82       \@@_int_if_zero_p:n \uproot@ && \@@_int_if_zero_p:n \leftroot@

```

```

83     }
84     {
85     \Uroot \c_@@_radical_sqrt_tl { #1 } { #2 }
86     }
87     {
88     \hbox_set:Nn \rootbox
89     {
90     \c_math_toggle_token \m@th
91     \scriptscriptstyle { #1 }
92     \c_math_toggle_token
93     }
94     \mathchoice
95     { \r@@@t \displaystyle { #2 } }
96     { \r@@@t \textstyle { #2 } }
97     { \r@@@t \scriptstyle { #2 } }
98     { \r@@@t \scriptscriptstyle { #2 } }
99     }
100    \c_group_end_token
101  }
102  </LU>
103  \@@_check_and_fix:NNnnn \r@@@t \cs_set_nopar:Npn { #1 #2 }
104  {
105  \setboxz@h { $ \m@th #1 \sqrt{sign { #2 } } $ }
106  \dimen@ \ht\z@
107  \advance \dimen@ -\dp\z@
108  \setbox\@ne \hbox { $ \m@th #1 \mskip \uproot@ mu $ }
109  \advance \dimen@ by 1.667 \wd\@ne
110  \mkern -\leftroot@ mu
111  \mkern 5mu
112  \raise .6\dimen@ \copy\rootbox
113  \mkern -10mu
114  \mkern \leftroot@ mu
115  \boxz@
116  }
117  < *LU >
118  {
119  \hbox_set:Nn \l_tmpa_box
120  {
121  \c_math_toggle_token \m@th
122  #1 \mskip \uproot@ mu
123  \c_math_toggle_token
124  }
125  \Uroot \c_@@_radical_sqrt_tl
126  {
127  \box_move_up:nn { \box_wd:N \l_tmpa_box }
128  {
129  \hbox:n
130  {
131  \c_math_toggle_token \m@th

```

```

132         \mkern -\leftroot@ mu
133         \box_use:N \rootbox
134         \mkern \leftroot@ mu
135         \c_math_toggle_token
136     }
137 }
138 }
139 { #2 }
140 }
141 </LU>
142 <*XE>
143 {
144     \hbox_set:Nn \l_tmpa_box
145     {
146         \c_math_toggle_token \meth
147         #1 \sqrtsign { #2 }
148         \c_math_toggle_token
149     }
150     \hbox_set:Nn \l_tmpb_box
151     {
152         \c_math_toggle_token \meth
153         #1 \mskip \uproot@ mu
154         \c_math_toggle_token
155     }
156     \mkern -\leftroot@ mu
157     \@_mathstyle_scale:NnnN #1 { \kern } { \fontdimen 63 \g_@@_sqrt_font_cmd_tl } \g_@@_sqrt_font_
158     \box_move_up:nn
159     {
160         \box_wd:N \l_tmpb_box + (\box_ht:N \l_tmpa_box - \box_dp:N \l_tmpa_box)
161         * \number \fontdimen 65 \g_@@_sqrt_font_cmd_tl / 100
162     }
163     { \box_use:N \rootbox }
164     \@_mathstyle_scale:NnnN #1 { \kern } { \fontdimen 64 \g_@@_sqrt_font_cmd_tl } \g_@@_sqrt_font_
165     \mkern \leftroot@ mu
166     \box_use_drop:N \l_tmpa_box
167 }
168 </XE>
169 }
170 </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 }
```

### 23.2 Unicode radicals

Make sure `\Uroot` is defined in the case where the  $\LaTeX$  kernel doesn't make it available with its native name.

`\@@_redefine_radical:`

```
19 \AtBeginDocument{ \ifpackageloaded { amsmath } { } { \@@_redefine_radical: } }
```

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

#2 : Leading superscript for the sqrt sign

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

20 *(\*XE)*

```
21 \cs_new:Nn \@@_redefine_radical:
```

```

22 {
23   \cs_set:nopar:Npn \r@@@t ##1 ##2
24   {
25     \hbox_set:Nn \l_tmpa_box
26     {
27       \c_math_toggle_token \m@th
28       ##1 \sqrtsign { ##2 }
29       \c_math_toggle_token
30     }
31     \@_mathstyle_scale:NnnN ##1 { \kern } { \fontdimen 63 \g_@@_sqrt_font_cmd_tl } \g_@@_sqrt_fo
32     \box_move_up:nn
33     {
34       (\box_ht:N \l_tmpa_box - \box_dp:N \l_tmpa_box) * \number \font-
35       dimen 65 \g_@@_sqrt_font_cmd_tl / 100
36     }
37     { \box_use:N \rootbox }
38     \@_mathstyle_scale:NnnN ##1 { \kern } { \fontdimen 64 \g_@@_sqrt_font_cmd_tl } \g_@@_sqrt_fo
39     \box_use_drop:N \l_tmpa_box
40   }
41 }

```

`\root` Redefine this macro for Lua $\TeX$ , which provides us a nice primitive to use.

```

42 (*LU)
43 \cs_new:Nn \@@_redefine_radical:
44 {
45   \cs_set:Npn \root ##1 \of ##2
46   {
47     \Uroot \c_@@_radical_sqrt_tl { ##1 } { ##2 }
48   }
49 }
50 </LU>

```

### 23.2.1 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  $\LaTeX$  fraction declaration.

```

51 \cs_new:Nn \@@_which_frac:nn
52 {
53   \bool_if:NTF \l_@@_smallfrac_bool {\tfrac} {\frac} {#1} {#2}
54 }
55 \cs_new:Npn \@@_setup_active_frac:
56 {
57   \@_mathactive_remap:nn {"2189} { \@@_which_frac:nn {0} {3} }
58   \@_mathactive_remap:nn {"2152} { \@@_which_frac:nn {1} {10} }
59   \@_mathactive_remap:nn {"2151} { \@@_which_frac:nn {1} {9} }
60   \@_mathactive_remap:nn {"215B} { \@@_which_frac:nn {1} {8} }

```

```

61 \@@_mathactive_remap:nn {"2150} { \@@_which_frac:nn {1} {7} }
62 \@@_mathactive_remap:nn {"2159} { \@@_which_frac:nn {1} {6} }
63 \@@_mathactive_remap:nn {"2155} { \@@_which_frac:nn {1} {5} }
64 \@@_mathactive_remap:nn {"00BC} { \@@_which_frac:nn {1} {4} }
65 \@@_mathactive_remap:nn {"2153} { \@@_which_frac:nn {1} {3} }
66 \@@_mathactive_remap:nn {"215C} { \@@_which_frac:nn {3} {8} }
67 \@@_mathactive_remap:nn {"2156} { \@@_which_frac:nn {2} {5} }
68 \@@_mathactive_remap:nn {"00BD} { \@@_which_frac:nn {1} {2} }
69 \@@_mathactive_remap:nn {"2157} { \@@_which_frac:nn {3} {5} }
70 \@@_mathactive_remap:nn {"215D} { \@@_which_frac:nn {5} {8} }
71 \@@_mathactive_remap:nn {"2154} { \@@_which_frac:nn {2} {3} }
72 \@@_mathactive_remap:nn {"00BE} { \@@_which_frac:nn {3} {4} }
73 \@@_mathactive_remap:nn {"2158} { \@@_which_frac:nn {4} {5} }
74 \@@_mathactive_remap:nn {"215A} { \@@_which_frac:nn {5} {6} }
75 \@@_mathactive_remap:nn {"215E} { \@@_which_frac:nn {7} {8} }
76 }
77 \AtBeginDocument { \@@_setup_active_frac: }

```

### 23.3 *Synonyms and all the rest*

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

```

78 \protected\def\to{\rightarrow}
79 \protected\def\le{\leq}
80 \protected\def\ge{\geq}
81 \protected\def\neq{\neq}
82 \protected\def\triangle{\mathord{\bigtriangleup}}
83 \protected\def\bigcirc{\mdlgwhtcircle}
84 \protected\def\circ{\vysmwhtcircle}
85 \protected\def\bullet{\smbllkcircle}
86 \protected\def\mathyen{\yen}
87 \protected\def\mathsterling{\sterling}
88 \protected\def\diamond{\smwhtdiamond}
89 \protected\def\emptyset{\varnothing}
90 \protected\def\hbar{\hslash}
91 \protected\def\land{\wedge}
92 \protected\def\lor{\vee}
93 \protected\def\owns{\ni}
94 \protected\def\gets{\leftarrow}
95 \protected\def\mathring{\ocirc}
96 \protected\def\lnot{\neg}
97 \protected\def\longdivision{\longdivisionsign}

```

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

```

98 \protected\def\backepsilon{\upbackepsilon}
99 \protected\def\eth{\matheth}

```

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

```

100 \protected\def\dbkarow {\dbkarrow}
101 \protected\def\drbkarow{\drbkarow}
102 \protected\def\hksearrow{\hksearrow}
103 \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:

```

104 \protected\def\smallint{\mathop{\textstyle\int}\limits}

```

`\underbar`

```

105 \cs_set_eq:NN \latexe_underbar:n \underbar
106 \renewcommand\underbar
107   {
108     \mode_if_math:TF \mathunderbar \latexe_underbar:n
109   }

```

`\colon` Define `\colon` as a mathpunct ':'. This is wrong; it should be `u+003A colon` instead! We hope no-one will notice.

```

110 \ifpackageloaded{amsmath}
111   {
112     % define their own colon, perhaps I should just steal it. (It does look much bet-
113     % ter.)
114   }
115   {
116     \cs_set_protected:Npn \colon
117       {
118         \bool_if:NTF \g_@@_literal_colon_bool {;} { \mathpunct{:} }
119       }

```

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

```

\Digamma
120 \protected\def\digamma{\updigamma}
121 \protected\def\Digamma{\upDigamma}

```

### *Symbols*

```

122 \cs_set_protected:Npn \l {\Vert}
      \mathinner items:
123 \cs_set_protected:Npn \mathellipsis {\mathinner{\unicodeellipsis}}
124 \cs_set_protected:Npn \cdots {\mathinner{\unicodcdots}}
125 \cs_set_eq:NN \@@_text_slash: \slash
126 \cs_set_protected:Npn \slash
127   {
128     \mode_if_math:TF {\mathslash} {\@@_text_slash:}
129   }

```



### 23.3.1 \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 Xe<sub>La</sub>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 Gregorio<sup>3</sup>.

\not

```
130 \DeclareDocumentCommand \not {m}
131 {
132   \tl_set:Nx \l_@@_not_token_name_tl { \cs_to_str:N #1 }
133   \tl_if_empty:NT \l_@@_not_token_name_tl
134   {
135     \tl_set:Nx \l_@@_not_token_name_tl { \token_to_str:N #1 }
136   }
137   \cs_if_exist:cTF { not \l_@@_not_token_name_tl }
138   {
139     \use:c { not \l_@@_not_token_name_tl }
140   }
141   {
142     \cs_if_exist:cTF { n \l_@@_not_token_name_tl }
143     {
144       \use:c { n \l_@@_not_token_name_tl }
145     }
146     {
147       \tl_if_eq:nnTF {#1} {$} { \notaccent{} } { \notaccent } #1
148     }
149   }
150 }
```

\NewNegationCommand

\RenewNegationCommand

```
151 \DeclareDocumentCommand \NewNegationCommand {mm}
152 {
153   \@@_set_negation_command:Nnn \cs_new_protected:cpn {#1} {#2}
154 }
155 \DeclareDocumentCommand \RenewNegationCommand {mm}
156 {
157   \@@_set_negation_command:Nnn \cs_set_protected:cpn {#1} {#2}
158 }
159 \cs_set:Nn \@@_set_negation_command:Nnn
```

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

```

160 {
161   \tl_set:Nx \l_@@_not_token_name_tl { \cs_to_str:N #2 }
162   \tl_if_empty:NT \l_@@_not_token_name_tl
163   {
164     \tl_set:Nx \l_@@_not_token_name_tl { \token_to_str:N #2 }
165   }
166   #1 { not \l_@@_not_token_name_tl } { #3 }
167 }

168 \NewNegationCommand { = } { \neq }
169 \NewNegationCommand { < } { \less }
170 \NewNegationCommand { > } { \ngtr }
171 \NewNegationCommand { \gets } { \leftarrow }
172 \NewNegationCommand { \simeq } { \sime }
173 \NewNegationCommand { \equal } { \ne }
174 \NewNegationCommand { \le } { \leq }
175 \NewNegationCommand { \ge } { \geq }
176 \NewNegationCommand { \greater } { \ngtr }
177 \NewNegationCommand { \forksnot } { \forks }

```

### 23.3.2 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

```

178 \int_step_inline:nnnn {0} {1} {9}
179 {
180   \@@_mathactive_remap:nn {"FF10+#1} {\char\int_eval:n{\0+#1}}
181 }

```

#### Letters

```

182 \int_step_inline:nnnn {0} {1} {26}
183 {
184   \@@_mathactive_remap:nn {"FF21+#1} {\char\int_eval:n{\A+#1}}
185   \@@_mathactive_remap:nn {"FF41+#1} {\char\int_eval:n{\a+#1}}
186 }

```

## 23.4 Legacy characters

\@@\_undeclare\_symbol:N

```

187 \cs_new:Nn \@@_undeclare_symbol:N
188 {
189   \cs_set_protected:Npn #1

```

```
190     { \@@_error:nx {legacy-char-not-supported} { \token_to_str:N #1 } }  
191   }
```

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

```
192 \@@_undeclare_symbol:N \arrowvert  
193 \@@_undeclare_symbol:N \Arrowvert  
194 \@@_undeclare_symbol:N \bracevert
```

## 24 *A secret hook*

This will be executed after most if not all of the standard unicode-math setup.

```
195 \AtBeginDocument{\g_@@_secret_hook_tl}  
196 </package>
```

## *Fin*

The official end of the package:

```
197 <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.

<b>Symbols</b>	
$\backslash$ \$	36
$\backslash$ &	37
$\backslash$ '	35, 260, 294
$\backslash$ *	269
$\backslash$ -	6, 268
$\backslash$ .	253
$\backslash$ /	321
$\backslash$ :	272
$\backslash$ :::	8
$\backslash$ ::N	8
$\backslash$ ::x_unbraced	8
$\backslash$ <	325
$\backslash$ =	7
$\backslash$ >	326
$\backslash$ @@@math@bgroup	53
$\backslash$ @@@math@egroup	54
$\backslash$ @_accent:nnn	<u>57</u> , 131, 138
$\backslash$ @_alphabet_config:nnn	123, 129, 141, 169
$\backslash$ @_arg_i_before_egroup:n	2, 6
$\backslash$ @_assign_delcode:n	<u>318</u> , 324, 327–353
$\backslash$ @_assign_delcode:nn	45, 69, <u>307</u> , 318, 321–323, 325, 326
$\backslash$ @_assign_delcode_noparse:nn	69, 307, 315
$\backslash$ @_assign_delcode_parse:nn	45, 311
$\backslash$ @_backprime_double_mchar	48, 256, 292
$\backslash$ @_backprime_single_mchar	168, 174, 180, 185, 190, 195, 255, 291
$\backslash$ @_backprime_triple_mchar	53, 257, 293
$\backslash$ @_char_gmake_mathactive:N	61
$\backslash$ @_char_gmake_mathactive:n	61, 73, 75, 305
$\backslash$ @_check_and_fix:NNnnn	<u>2</u> , 22, 43, 66, 103
$\backslash$ @_check_math_alphabet:	79, <u>83</u>
$\backslash$ @_copy_fontdimen:nnN	<u>81</u> , 199–211, 233–237
$\backslash$ @_cs_new:Nn	2, 6, 10, 12, 19, 23, 24, 29, 30, 32, 38, 39, 44, 48, 73, 75, 85, 110, <u>136</u> , 154, 176, 191, 226, 251, 266, 285, 319, 365, 370
$\backslash$ @_cs_set:Nn	60, 69, 78, 87, 93, 99, 104, 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, <u>72</u>
$\backslash$ @_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	<u>53</u> , 109, 117, 124, 126
$\backslash$ @_error:n	21, 78, 127
$\backslash$ @_error:nx	112, 128, 190
$\backslash$ @_font_dimen:Nnnnn	<u>12</u> , 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	<u>73</u>
$\backslash$ @_font_param_aux:NNnnnn	<u>73</u>
$\backslash$ @_font_param_aux:ccc	58
$\backslash$ @_font_param_aux:ccnnnn	50
$\backslash$ @_fontdimen_from_param:Nnn	89
$\backslash$ @_fontdimen_from_param:nn	90, 156–173
$\backslash$ @_fontdimen_pc_to_pt:nN	73, 74, <u>146</u>
$\backslash$ @_fontdimen_to_percent:nN	<u>146</u> , 157, 160
$\backslash$ @_fontface_gset_eq:NN	24, 100, 146
$\backslash$ @_fontface_gset_eq:cN	142
$\backslash$ @_fontspec_select_font:	15, <u>110</u>
$\backslash$ @_fontspec_trial_font:	12, <u>85</u>
$\backslash$ @_fontswitch:n	43, <u>45</u>
$\backslash$ @_glyph_if_exist:Nn	16
$\backslash$ @_glyph_if_exist:NnT	96
$\backslash$ @_glyph_if_exist:NnTF	16, 22, 27, 32, 47, 52, 126
$\backslash$ @_group_begin:	8, 12, 14, 48, 51
$\backslash$ @_group_begin_frozen:	14, 51
$\backslash$ @_group_end:n	13, 15, 16, 49, 52, 56
$\backslash$ @_group_end_frozen:n	15, 52



$\@@_scan\_qprime$ : . . . . . 79, 141  
 $\@@_scan\_sscript$ : . . . . . 10, 21, 24, 27  
 $\@@_scan\_sscript:TF$  . . . . . 26, 30  
 $\@@_scan\_sscript\_aux:nnN$  . . . . . 35, 39  
 $\@@_scan\_sup\_backdprime$ : . . . . . 187, 278  
 $\@@_scan\_sup\_backprime$ : . . 182, 276, 277  
 $\@@_scan\_sup\_backtrprime$ : . . . 192, 279  
 $\@@_scan\_sup\_dprime$ : . . . . . 90, 273  
 $\@@_scan\_sup\_prime$ : . . . . . 85, 271, 272  
 $\@@_scan\_sup\_qprime$ : . . . . . 100, 275  
 $\@@_scan\_sup\_trprime$ : . . . . . 95, 274  
 $\@@_scan\_trprime$ : . . . . . 73, 129  
 $\@@_scanbackprime\_collect:N$  . . . . .  
. . . 168, 174, 180, 185, 190, 195,  
197, 202, 207, 212, 218, 224, 230, 236  
 $\@@_scanprime\_collect:N$  . . . . . 65,  
71, 77, 83, 88, 93, 98, 103, 105, 109,  
112, 115, 120, 126, 132, 138, 144, 150  
 $\@@_set\_big\_operator:nnn$  . . . . . 45, 73  
 $\@@_set\_delcode:nnn$  . . . . .  
. . . . . 45, 106, 114, 122, 253, 309  
 $\@@_set\_math\_accent:Nnnn$  . . . . .  
. . . . . 50, 52, 54, 56, 62, 63, 128  
 $\@@_set\_math\_close:nnn$  . . . . . 47, 112  
 $\@@_set\_math\_fence:nnnn$  . . . . . 48, 119  
 $\@@_set\_math\_open:nnn$  . . . . . 46, 87  
 $\@@_set\_math\_overunder:Nnnn$  58, 60, 133  
 $\@@_set\_mathalph\_range:nnnn$  . 289, 294  
 $\@@_set\_mathalph\_range:nnnnn$  . . . . .  
. . . . . 294, 316, 321, 327, 335, 343  
 $\@@_set\_mathalphabet\_Greek:nnn$  . .  
. . . . . 42–44,  
127, 259, 331, 378, 382, 388, 501,  
505, 511, 647, 653, 656, 760, 766, 769  
 $\@@_set\_mathalphabet\_Latin:nnn$  . .  
. . . . . 14–16,  
94, 174, 208, 223, 228, 240, 261,  
267, 270, 294, 300, 303, 318, 328,  
340, 344, 350, 463, 467, 473, 584,  
593, 601, 613, 619, 622, 726, 732, 735  
 $\@@_set\_mathalphabet\_char:nnn$  . . .  
. . . . . 18, 27, 59, 270, 286, 292  
 $\@@_set\_mathalphabet\_char:nnnn$  . .  
. . . . . 284, 307, 328, 336, 344–349  
 $\@@_set\_mathalphabet\_greek:nnn$  . .  
. . . . . 56–58,  
139, 239, 339, 395, 399, 405, 518,  
522, 528, 664, 670, 673, 777, 783, 786  
 $\@@_set\_mathalphabet\_latin:nnn$  . .  
. . 31–33, 113, 169, 221, 231, 249,  
278, 284, 287, 311, 317, 320, 323,  
332, 361, 365, 371, 484, 488, 494,  
589, 597, 630, 636, 639, 743, 749, 752  
 $\@@_set\_mathalphabet\_numbers:nnn$  .  
. . . . . 5, 186, 253,  
254, 267, 313, 324, 453, 454, 605, 606  
 $\@@_set\_mathalphabet\_pos:nnnn$  . . .  
. . . . . 83–86, 114, 115, 164,  
165, 175–181, 191–195, 200–204,  
209–216, 222–224, 229–236, 241–  
245, 247–252, 262, 302, 426, 427,  
431, 436, 442, 447, 555–560, 563,  
568, 573, 578, 698, 699, 702, 707,  
712, 717, 811, 812, 815, 820, 825, 830  
 $\@@_set\_mathchar:NNnn$  . . . . . 39, 44, 304  
 $\@@_set\_mathchar:cNnn$  . . . . . 39, 79  
 $\@@_set\_mathcode:nnn$  . . . . .  
. . . . . 29, 40–44, 107, 115, 121  
 $\@@_set\_mathcode:nnnn$  29, 180, 274, 283  
 $\@@_set\_mathsymbol:nNnn$  . . . . . 27, 257  
 $\@@_set\_negation\_command:Nnn$  . . . . .  
. . . . . 153, 157, 159  
 $\@@_set\_normal\_Greek:nn$  . . . . .  
. . . . . 38, 40, 122, 125, 255,  
381, 387, 504, 510, 646, 652, 759, 765  
 $\@@_set\_normal\_Latin:nn$  . . . . .  
. . . . . 10, 12, 90, 92,  
219, 260, 266, 293, 299, 338, 343,  
349, 461, 466, 472, 612, 618, 725, 731  
 $\@@_set\_normal\_char:nnn$  . . . . .  
. . 26–28, 65, 70, 75, 80, 102, 108–  
110, 146, 151, 156, 161, 205, 413,  
416, 420, 423, 537, 542, 547, 552,  
680, 685, 690, 695, 793, 798, 803, 808  
 $\@@_set\_normal\_greek:nn$  . . . . .  
. . . . . 49, 53, 134, 137, 235,  
398, 404, 521, 527, 663, 669, 776, 782  
 $\@@_set\_normal\_latin:nn$  . . . . .  
. . . . . 21, 25, 101, 107,  
227, 277, 283, 310, 316, 359, 364,  
370, 482, 487, 493, 629, 635, 742, 748  
 $\@@_set\_normal\_numbers:nn$  . . . . . 4, 265  
 $\@@_setmathfont:nn$  . . . . . 2, 5  
 $\@@_setmathfontface:Nnn$  . . . . . 2, 9  
 $\@@_setup\_active\_frac$ : . . . . . 55, 77  
 $\@@_setup\_active\_subscript:nn$  . . .  
. . . . . 13, 117–153



<code>\beta</code> .....	111, 149		
<code>\bfdefault</code> .	91, 96, 102, 107, 188, 223, 248		
<code>\bgroup</code> .....	5, 25, 34		
<code>\bigcirc</code> .....	83		
<code>\bigtriangleright</code> .....	82		
<code>\bool_gset_false:N</code> .....	22, 28, 33, 38, 43, 48, 53, 59–63, 66, 68–70, 73, 77, 80, 104, 105, 116–120, 123, 125, 130, 142, 149, 153, 154, 163, 167, 168, 177		
<code>\bool_gset_true:N</code> .....	29, 34, 35, 39, 44, 49, 51, 54, 60, 67, 74–76, 81–84, 87, 124, 126, 127, 131–134, 137, 143, 144, 150, 157, 164, 171, 176		
<code>\bool_if:NF</code> .....	.. 21, 43, 92, 105, 125, 137, 149, 159, 270, 297, 314, 336, 347, 357, 368, 385, 402, 415, 422, 434, 445, 729, 746, 763, 780, 796, 806, 818, 828		
<code>\bool_if:NT</code>	12, 13, 17, 23, 25, 40, 51, 68, 78, 144, 264, 281, 459, 470, 480, 491, 508, 525, 540, 550, 566, 576, 616, 633, 650, 667, 683, 693, 705, 715		
<code>\bool_if:NTF</code> .....	. 4, 10, 21, 33, 38, 49, 53, 63, 72– 80, 90, 99, 117, 120, 132, 144, 154, 258, 275, 291, 308, 341, 362, 379, 396, 412, 419, 429, 440, 464, 485, 502, 519, 535, 545, 561, 571, 610, 627, 644, 661, 678, 688, 700, 710, 723, 740, 757, 774, 791, 801, 813, 823		
<code>\bool_if:nTF</code> .....	80		
<code>\bool_lazy_and:nnT</code> .....	29		
<code>\bool_lazy_and:nnTF</code> .....	103		
<code>\bool_new:N</code> .....	2–28		
<code>\bool_set_false:N</code> .....	26, 193, 200		
<code>\bool_set_true:N</code> .....	14, 190		
<code>\box</code> .....	64		
<code>\box_dp:N</code> .....	34, 160		
<code>\box_ht:N</code> .....	34, 77, 160		
<code>\box_move_up:nn</code> .....	32, 127, 158		
<code>\box_set_ht:Nn</code> .....	75		
<code>\box_use:N</code> .....	36, 133, 163		
<code>\box_use_drop:N</code> .....	38, 79, 166		
<code>\box_wd:N</code> .....	127, 160		
<code>\boxz@</code> .....	115		
<code>\bracket</code> .....	194		
<code>\bullet</code> .....	85		
		<b>C</b>	
<code>\C</code> .....	139		
<code>\c_@_math_alphabet_name_Greek_tl</code>	36		
<code>\c_@_math_alphabet_name_greek_tl</code>	35		
<code>\c_@_math_alphabet_name_Latin_tl</code>	34		
<code>\c_@_math_alphabet_name_latin_tl</code>	33		
<code>\c_@_math_alphabet_name_misc_tl</code> .	38		
<code>\c_@_math_alphabet_name_num_tl</code> ..	37		
<code>\c_@_radical_sqrt_tl</code> .....	47, 85, 125		
<code>\c_group_begin_token</code> .....	42, 54		
<code>\c_group_end_token</code> .....	100		
<code>\c_math_toggle_token</code> .....	.. 27, 29, 56, 60, 69, 73, 90, 92, 121, 123, 131, 135, 146, 148, 152, 154		
<code>\c_parameter_token</code> .....	59		
<code>\c_space_tl</code> .....	85, 86		
<code>\calculate@math@sizes</code> .....	7		
<code>\cdots</code> .....	15, 124		
<code>\cdp@elt</code> .....	110		
<code>\cdp@list</code> .....	110		
<code>\char</code> .....	180, 184, 185		
<code>\char_generate:nn</code> .....	21		
<code>\char_gset_active_eq:nN</code> .....	74, 77		
<code>\char_set_catcode_active:N</code> ..	260, 261		
<code>\char_set_catcode_active:n</code> ..	262–268		
<code>\char_set_catcode_ignore:n</code> .....	116		
<code>\char_set_catcode_other:n</code> .....	36		
<code>\char_set_catcode_space:n</code> .....	2		
<code>\char_value_catcode:n</code> .....	34		
<code>\chi</code> .....	115, 153		
<code>\circ</code> .....	84		
<code>\cirfnint</code> .....	359		
<code>\clist_clear:N</code> .....	62		
<code>\clist_gclear_new:c</code> .....	162		
<code>\clist_gput_right:cx</code> .	197, 214, 296, 309		
<code>\clist_gput_right:Nx</code> .....	84–86		
<code>\clist_gremove_duplicates:c</code> .....	165		
<code>\clist_if_empty:NF</code> .....	7		
<code>\clist_if_empty:NT</code> .....	72		
<code>\clist_if_exist:cT</code> .....	42		
<code>\clist_if_in:NnT</code> .....	279		
<code>\clist_if_in:NVT</code> .....	78		
<code>\clist_map_break:</code> .....	51, 93, 100		
<code>\clist_map_break:n</code> .....	51		
<code>\clist_map_inline:cn</code> .....	44		
<code>\clist_map_inline:Nn</code> ...	40, 46, 85, 112		
<code>\clist_map_inline:nn</code> ...	5, 52, 53, 61, 64, 75, 208, 209, 221, 229, 237, 257, 282, 306, 315, 320, 325, 333, 341		



<code>\clist_new:N</code> .....	76–83	<code>\cs_new:Npn</code> .....	3, 55, 61, 67, 73, 79, 85, 90, 95, 100, 127–132, 152, 162, 164, 170, 176, 182, 187, 192
<code>\clist_put_right:Nx</code> .....	142	<code>\cs_new_eq:Nc</code> .....	6
<code>\clist_set:No</code> .....	38, 68	<code>\cs_new_nopar:Npn</code> .....	13, 76, 90
<code>\Colon</code> .....	117, 126	<code>\cs_new_protected:cpn</code> .....	153
<code>\colon</code> .....	<u>110</u>	<code>\cs_new_protected:Nn</code> 25, 77, 90, 101, 164	
<code>\coloncolon</code> .....	78, <u>122</u>	<code>\cs_new_protected_nopar:Nn</code> .....	2, 47, 62, 66, 70, 74, 88
<code>\coloncolonequals</code> .....	82, <u>122</u>	<code>\cs_new_protected_nopar:Npn</code> ...	80, 94
<code>\Coloneq</code> .....	119, 130	<code>\cs_set:cpx</code> .....	70
<code>\coloneq</code> .....	118, 128	<code>\cs_set:Nn</code> .....	27, 45, 63, 146, 159, 192, 205, 255, 259, 381
<code>\Coloneqq</code> .....	69, <u>114</u>	<code>\cs_set:Npn</code> .....	4, 14, 15, 22, 27, 29, 31, 32, 34, 39, 40, 45, 49, 91, 92, 96, 100, 151, 166
<code>\coloneqq</code> .....	68, <u>114</u>	<code>\cs_set_eq:cc</code> .....	63
<code>\colonequals</code> .....	80, <u>122</u>	<code>\cs_set_eq:NN</code> .....	6, 8, 12, 14, 15, 18, 19, 27, 28, 42– 48, 51–54, 56, 59, 60, 63, 66–70, 74, 75, 81, 85, 86, 105, 108–111, 125, 166, 172, 178, 251–257, 271–279
<code>\coloneqsep</code> .....	85	<code>\cs_set_nopar:Npn</code> .....	23, 43, 66, 103
<code>\color@endgroup</code> .....	73	<code>\cs_set_protected:cpn</code> .....	23, 157
<code>\color@setgroup</code> .....	69	<code>\cs_set_protected:cpx</code> .....	6, 78
<code>\copy</code> .....	112	<code>\cs_set_protected:Nn</code> .....	24, 155
<code>\crampeddisplaystyle</code> .....	<u>8</u> , 18	<code>\cs_set_protected:Npn</code> .....	13, 72, 86, 87, 115, 122–124, 126, 189
<code>\crampedscriptscriptstyle</code> .....	<u>8</u>	<code>\cs_set_protected_nopar:Npn</code> .....	8
<code>\crampedscriptstyle</code> .....	<u>8</u> , 30	<code>\cs_set_protected_nopar:Npx</code> .....	76
<code>\crampedtextstyle</code> .....	<u>8</u> , 24	<code>\cs_to_str:N</code> .....	6, 76, 79, 81, 83, 94, 96, 100, 102, 123, 125, 132, 157, 159, 161, 166, 168
<code>\crrc</code> .....	38, 62	<code>\csname</code> .....	32, 37, 42, 47, 51, 55
<code>\cs_generate_variant:Nn</code> .....	9–11, 28, 44, 59, 85, 99, 133, 134	<code>\cuberoot</code> .....	363
<code>\cs_gset:cpx</code> .....	81		
<code>\cs_gset:Nn</code> .....	269		
<code>\cs_gset_eq:NN</code> .....	21, 27–29		
<code>\cs_gset_protected_nopar:cpx</code> .....	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> 10, 15, 18, 21, 24, 27, 30, 33			
<code>\cs_if_exist:cF</code> .....	7, 178		
<code>\cs_if_exist:cTF</code> ....	105, 137, 142, 175		
<code>\cs_if_exist:NF</code> .....	5, 91, 154		
<code>\cs_if_exist:NT</code> .....	4, 222		
<code>\cs_if_exist:NTF</code> .....	189		
<code>\cs_new:cn</code> .....	160		
<code>\cs_new:Nn</code> .....	2, 3, 8, 13– 16, 19, 21, 41–43, 45, 49, 51, 53, 57, 58, 61, 65, 69, 73, 74, 81, 83, 85, 87, 101, 102, 105, 112, 119, 128, 133, 135, 138, 142, 147, 148, 150, 154, 162, 169, 178, 182, 187, 197, 200, 219, 227, 235, 249, 255, 265, 270, 275, 277, 280, 284, 289, 294, 297, 302, 307, 311, 313, 318, 323, 331, 339		

## D



**G**

<code>\g</code> .....	168	<code>\g_@@_literal_Nabla_bool</code> .....	20, 63, 144, 149, 153, 157, 412, 440, 535, 571, 678, 710, 791, 823
<code>\g_@@_alphabet_clist</code> .....	78	<code>\g_@@_literal_partial_bool</code> .....	21, 73, 154, 163, 167, 171, 419, 429, 545, 561, 688, 700, 801, 813
<code>\g_@@_bad_alpha_clist</code> .....	78, 83–86	<code>\g_@@_main_font_cmd_tl</code> .....	72, 150, 199–211, 233–237
<code>\g_@@_bfit_Greek_usv</code> .....	78	<code>\g_@@_main_font_defined_bool</code> ....	3, 21, 35, 43
<code>\g_@@_bfit_greek_usv</code> .....	79	<code>\g_@@_mathalph_seq</code> .....	6, 13, 34, 49, 50, 63, 65, 82, 88
<code>\g_@@_bfit_Latin_usv</code> .....	76	<code>\g_@@_mathbf_text_bool</code> ....	26, 43, 44
<code>\g_@@_bfit_latin_usv</code> .....	77	<code>\g_@@_mathclasses_seq</code> .....	25, <u>91</u>
<code>\g_@@_bfliteral_bool</code> .....	11, 116, 123, 130, 137, 341, 362, 379, 396, 464, 485, 502, 519	<code>\g_@@_mathit_text_bool</code> ....	25, 38, 39
<code>\g_@@_bfsfit_Greek_usv</code> .....	74	<code>\g_@@_mathparam_settings_tl</code>	59, 368, 373
<code>\g_@@_bfsfit_greek_usv</code> .....	75	<code>\g_@@_mathparam_store_tl</code> .....	<u>73</u>
<code>\g_@@_bfsfit_Latin_usv</code> .....	72	<code>\g_@@_mathrm_text_bool</code> 24, 28, 29, 33, 34	
<code>\g_@@_bfsfit_latin_usv</code> .....	73	<code>\g_@@_mathsf_text_bool</code> ....	27, 48, 49
<code>\g_@@_bfsfup_Greek_usv</code> .....	74	<code>\g_@@_mathstyles_seq</code> .....	4, <u>101</u>
<code>\g_@@_bfsfup_greek_usv</code> .....	75	<code>\g_@@_mathtable_tl</code> .....	8, 10, 61
<code>\g_@@_bfsfup_Latin_usv</code> .....	72	<code>\g_@@_mathtt_text_bool</code> ....	28, 53, 54
<code>\g_@@_bfsfup_latin_usv</code> .....	73	<code>\g_@@_mclass_range_seq</code> 48, 90, 109, 122	
<code>\g_@@_bfup_Greek_usv</code> .....	78	<code>\g_@@_mversion_tl</code> .....	63
<code>\g_@@_bfup_greek_usv</code> .....	79	<code>\g_@@_operator_mathfont_tl</code> .	14, 43, 57
<code>\g_@@_bfup_Latin_usv</code> .....	76	<code>\g_@@_ot_math_bool</code> .....	2, 51, 104
<code>\g_@@_bfup_latin_usv</code> .....	77	<code>\g_@@_prime_font_cmd_tl</code> .....	22, 27, 32, 47, 52, 72, 152
<code>\g_@@_bfupGreek_bool</code> .....	14, 78, 117, 124, 131, 385, 508	<code>\g_@@_primekern_muskip</code> ...	13, 104, 105
<code>\g_@@_bfupgreek_bool</code> .....	15, 79, 118, 125, 132, 402, 525	<code>\g_@@_remap_style_tl</code> .....	68
<code>\g_@@_bfupLatin_bool</code> .....	12, 76, 119, 126, 133, 336, 347, 459, 470	<code>\g_@@_secret_hook_tl</code> .....	32, 195
<code>\g_@@_bfuplatin_bool</code> .....	13, 77, 120, 127, 134, 357, 368, 480, 491	<code>\g_@@_sfliteral_bool</code> .....	17, 144, 258, 275, 291, 308, 610, 627, 644, 661, 723, 740, 757, 774
<code>\g_@@_char_nrange_clist</code> .....	76	<code>\g_@@_size_sfssf_fp</code> ..	78, 107, 123, 128
<code>\g_@@_char_range_seq</code> .....	47, 61, 89, 110, 116, 126, 129	<code>\g_@@_size_tfsf_fp</code> ...	77, 106, 120, 123
<code>\g_@@_curr_font_cmd_tl</code> .....	39, 70, 96, 126, 143	<code>\g_@@_slash_delimiter_usv</code> .....	58, 181–183, 321–323
<code>\g_@@_debug_nest_int</code> .	137, 140, 144, 149	<code>\g_@@_sqrt_font_cmd_tl</code> .....	31, 34, 37, 71, 151, 157, 161, 164
<code>\g_@@_default_mathalph_seq</code> .....	13, 32, 35, <u>100</u>	<code>\g_@@_style_tl</code> .....	67
<code>\g_@@_fam_int</code> .....	29, 40, 41	<code>\g_@@_subs_prop</code> .....	15, 103
<code>\g_@@_family_tl</code> .....	66	<code>\g_@@_supers_prop</code> .....	4, 102
<code>\g_@@_fontname_tl</code> .....	62	<code>\g_@@_symfont_tl</code> .....	64
<code>\g_@@_fonts_used_int</code> .	30, 139, 142, 143	<code>\g_@@_trial_font</code> .....	73, 74, 80, 100
<code>\g_@@_init_bool</code> .....	4, 12, 13, 17, 22, 25, 33, 60, 105, 144	<code>\g_@@_unknown_keys_clist</code> .....	77
<code>\g_@@_literal_bool</code> ...	6, 10, 21, 38, 49, 59, 66, 73, 80, 87, 90, 99, 120, 132	<code>\g_@@_upGreek_bool</code> 9, 40, 60, 67, 74, 81, 125	
<code>\g_@@_literal_colon_bool</code> .....	23, 117, 176, 177, 270	<code>\g_@@_upgreek_bool</code> .....	10, 51, 61, 68, 75, 82, 137



<b>K</b>	
\kern .....	31, 37, 157, 164
\keys_define:nn .....	2, 7, 10, 17, 204
\keys_set:nn .....	2
\keys_set_known:nnN .....	10
<b>L</b>	
\L .....	145
\l_@@_alphabet_clist .....	38, 40, 68, 72, 76, 82, 85, 112
\l_@@_alphabet_tl .....	41, 87, 88, 90, 97, 106, 114, 116, 118, 120, 122, 123, 126, 128, 129, 137, 141
\l_@@_curr_named_slot .....	47, 163, 197, 214, 296, 309
\l_@@_fam_three_tl ...	46, 228, 243, 248
\l_@@_fam_two_tl ...	45, 193, 218, 223
\l_@@_family_tl	40, 137, 141, 181, 184, 188
\l_@@_font .....	146, 150–152
\l_@@_font_keyval_tl .....	65, 87, 96, 112, 137, 195, 230
\l_@@_fontname_tl .....	10, 42, 50, 57, 58, 96, 137, 215, 240
\l_@@_implicit_alph_bool .	5, 14, 26, 132
\l_@@_keyval_clist .....	10, 13, 81
\l_@@_mathmap_charints_clist ....	7, 46, 62, 79, 142, 279
\l_@@_mathstyle_tl .....	4, 12, 51
\l_@@_mathtable_tl .....	3, 8, 60
\l_@@_missing_alph_seq .....	21, 30, 40, 64, 87, 134
\l_@@_mversion_tl .....	4, 6–8, 16, 18, 22, 52, 183, 185, 217, 220, 242, 245
\l_@@_nolimits_tl ...	19, 23, 53, 84, <u>355</u>
\l_@@_not_token_name_tl .....	69, 132, 133, 135, 137, 139, 142, 144, 161, 162, 164, 166
\l_@@_primecount_int .....	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
\l_@@_radicals_tl .....	52, 89, <u>363</u>
\l_@@_remap_style_tl	44, 69, 123, 126, 129
\l_@@_script_features_tl ...	12, 55, 125
\l_@@_script_font_tl .....	14, 57, 124
\l_@@_smallfrac_bool	22, 53, 190, 193, 200
\l_@@_ss_chain_tl .....	7, 18, 28, 46, 55
\l_@@_sscript_features_tl ..	13, 56, 130
\l_@@_sscript_font_tl .....	15, 58, 129
\l_@@_style_tl .....	37, 39, 42, 44, 67, 75, 88, 92, 97, 99, 106, 116, 118, 122, 123, 128, 129, 136, 141
\l_@@_symfont_label_tl .....	41, 43, 53, 178, 180, 183, 187, 257, 274, 283, 304, 309
\l_@@_tmpa_key_tl .....	41, 43, 56
\l_@@_tmpa_tl .....	9, 12, 19, 20, 22, 27–29, 42, 48, 56, 57, 59, 60, 64, 65, 67, 68, 70–73, 76, 78, 80, 84, 93, 98
\l_@@_tmpb_tl .....	44, 46, 49, 61, 85, 99
\l_@@_tmpc_tl .....	50, 62, 75, 76, 86, 94
\l_@@_trial_family_tl .....	54, 96, 99
\l_@@_unknown_keys_clist	10, 80, 93, 134
\l_MT_bracketheight_fdim .....	94, 102
\l_tmpa_box .....	25, 34, 38, 67, 75, 77, 79, 119, 127, 144, 160, 166
\l_tmpb_box .....	150, 160
\land .....	91
\latex_underbar:n .....	105, 108
\le .....	79, 174
\leaders .....	96, 104
\leftarrow .....	94
\leftroot@	82, 110, 114, 132, 134, 156, 165
\leq .....	79
\Let@ .....	26, 43
\limits .....	104, 140
\lineskip .....	31, 32, 51, 52
\lineskiplimit .....	32, 52
\lnot .....	96
\longdivision .....	97, 363
\longdivisionsign .....	97
\lor .....	92
\lowint .....	361
\lua_now:n .....	93
<b>M</b>	
\M .....	146
\m@th .....	27, 36, 45, 57, 69, 70, 90, 105, 108, 121, 131, 146, 152
\math .....	71
\math@bgroup .....	34, 47, 53
\math@egroup .....	38, 54, 56
\mathaccent .....	49, 59, 98
\mathaccentoverlay .....	61, 71, 98
\mathaccentwide .....	53, 70, 98
\mathalpha ..	18, 41, 94, 140, 180, 184, 274
\mathbacktick .....	295

<code>\mathbf</code>	91, 96	<code>\msg_redirect_module:nnn</code>	215, 216
<code>\mathbin</code>	18, 42, 94, 268, 269	<code>\msg_redirect_name:nnn</code>	209
<code>\mathbotaccent</code>	51, 69, 98	<code>\msg_warning:nn</code>	90, 116, 124, 129
<code>\mathbotaccentwide</code>	55, 72, 98	<code>\msg_warning:nxx</code>	24, 105
<code>\mathchar</code>	34–37	<code>\msg_warning:nxxx</code>	130
<code>\mathchar@type</code>	32, 37, 42, 55, 59, 121	<code>\msg_warning:nxxxx</code>	17
<code>\mathchardef</code>	6, 7	<code>\mskip</code>	13, 108, 122, 153
<code>\mathchoice</code>	72, 94	<code>\MT_cramped_internal:Nn</code>	43
<code>\mathclose</code>	47, 96, 115, 117, 126	<code>\MToverbracket</code>	85, 110
<code>\mathcode</code>	6, 7	<code>\MTunderbracket</code>	86, 111
<code>\mathellipsis</code>	123	<code>\muskip_gset:Nn</code>	105
<code>\matheth</code>	99	<code>\muskip_new:N</code>	104
<code>\mathfence</code>	18, 48, 66, 97		
<code>\mathinner</code>	14, 123, 124	<b>N</b>	
<code>\mathit</code>	90, 97	<code>\N</code>	147
<code>\mathop</code>	18, 45, 79, 95, 104, 137	<code>\ne</code>	81, 173
<code>\mathopen</code>	46, 96, 107, 109, 124	<code>\neg</code>	96
<code>\mathord</code>	18, 40, 82, 94, 287–295	<code>\neq</code>	81, 168
<code>\mathover</code>	57, 67, 97	<code>\new@mathalphabet</code>	117
<code>\mathpunct</code>	18, 44, 94, 117	<code>\new@mathgroup</code>	110
<code>\mathratio</code>	125	<code>\new@mathversion</code>	115
<code>\mathrel</code>	18, 43, 94, 272	<code>\new@symbolfont</code>	116
<code>\mathring</code>	95	<code>\NewDocumentCommand</code>	
<code>\mathrm</code>	16, 27, 86, 89, 95	.....	2, 3, 7, 12, 17, 21, 117–120
<code>\mathsf</code>	28, 101, 102	<code>\newmathalphabet</code>	111
<code>\mathslash</code>	128	<code>\newmathalphabet@</code>	111
<code>\mathsterling</code>	87	<code>\newmathalphabet@@</code>	111
<code>\mathstraightquote</code>	294	<code>\NewNegationCommand</code>	151, 168–177
<code>\Mathstrutbox@</code>	20	<code>\ngeq</code>	175
<code>\mathtt</code>	29, 106, 107	<code>\ngtr</code>	170, 176
<code>\mathunder</code>	59, 68, 97	<code>\ni</code>	93
<code>\mathunderbar</code>	108	<code>\nleftarrow</code>	171
<code>\mathup</code>	86	<code>\nleq</code>	174
<code>\mathyen</code>	86	<code>\nless</code>	169
<code>\mddefault</code>	19, 22, 89,	<code>\nolimits</code>	84
90, 95, 97, 101, 106, 181, 184, 218, 243		<code>\non@alpherr</code>	11
<code>\mdlgwhtcircle</code>	83	<code>\not</code>	130
<code>\minuscolon</code>	79, 122	<code>\notaccent</code>	147
<code>\mkern</code>	110, 111, 113, 114, 132, 134, 156, 165	<code>\npolint</code>	360
<code>\mode_if_math:F</code>	11	<code>\nsime</code>	172
<code>\mode_if_math:TF</code>	108, 128	<code>\nulldelimiterspace</code>	45, 71
<code>\msg_error:nn</code>	58, 127	<code>\number</code>	34, 161
<code>\msg_error:nxx</code>	128		
<code>\msg_info:nxx</code>	12	<b>O</b>	
<code>\msg_log:nn</code>	131	<code>\o</code>	172
<code>\msg_log:nxx</code>	132	<code>\ocirc</code>	95
<code>\msg_new:nnn</code>	56, 133	<code>\of</code>	45, 66
<code>\msg_new:nnnn</code>	134	<code>\oiint</code>	357
<code>\msg_new:nxx</code>	135	<code>\oint</code>	357
		<code>\oint</code>	357

<code>\ointctrlockwise</code> .....	358		
<code>\operator@font</code> .....	<u>40</u>		
<code>\overbracket</code> .....	56, <u>83</u>		
<code>\owns</code> .....	93		
<b>P</b>			
<code>\P</code> .....	148		
<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> .....	66		
<code>\pointint</code> .....	360		
<code>\prg_new_conditional:Nnn</code>	16, 97, 103, 173		
<code>\prg_replicate:nn</code> .....	13, 140		
<code>\prg_return_false:</code> ...	21, 99, 106, 176		
<code>\prg_return_true:</code> ...	19, 99, 106, 176		
<code>\prime</code> .....	251		
<code>\process@table</code> .....	114		
<code>\ProcessKeysOptions</code> .....	223		
<code>\prop_get:cnN</code> .....	75		
<code>\prop_get:cxNTF</code> .....	42		
<code>\prop_get:NnNTF</code> .....	10		
<code>\prop_gput:cnN</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> .....	102, 103		
<code>\protect</code> .....	5		
<code>\protected</code> .....	78–104, 120, 121		
<code>\ProvidesExplPackage</code> .....	41–43		
<b>Q</b>			
<code>\Q</code> .....	149		
<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, <u>108</u>		
<code>\qprime</code> .....	254		
<code>\ququad</code> .....	49		
<code>\quark_if_recursion_tail_stop:n</code>	21, 22		
<code>\quark_new:N</code> .....	108		
<b>R</b>			
<code>\R</code> .....	150		
<code>\r@@@t</code> .....	23, 73–76, 95–98, 103		
<code>\r@t</code> .....	20		
<code>\radical</code> .....	45		
<code>\raise</code> .....	112		
<code>\ratio</code> .....	77, <u>122</u>		
<code>\relax</code> .....	6, 7, 83, 87		
<code>\removenolimits</code> .....	<u>21</u>		
<code>\renewcommand</code> .....	106		
<code>\RenewDocumentCommand</code> .....	125–130		
<code>\RenewNegationCommand</code> .....	<u>151</u>		
<code>\RequirePackage</code> .....	3–7, 40		
<code>\RequirePackageWithOptions</code> ...	52, 53		
<code>\resetMathstrut@</code> .....	17		
<code>\restore@math@cr</code> .....	27, 44		
<code>\restore@mathversion</code> .....	114		
<code>\rho</code> .....	151		
<code>\rightarrow</code> .....	78		
<code>\rmdefault</code> .....	62, 109, 114		
<code>\root</code> .....	<u>42</u>		
<code>\rootbox</code> .....	36, 68, 88, 112, 133, 163		
<code>\rppolint</code> .....	359		
<b>S</b>			
<code>\sb</code> .....	19		
<code>\sbox</code> .....	45		
<code>\scan_stop:</code> ..	6, 18, 32, 37, 42, 47, 48,		
	51, 55, 57, 59, 63, 67, 79, 94, 383–427		
<code>\scpolint</code> .....	360		
<code>\scriptfont</code> .....	28–31, 55		
<code>\scriptscriptfont</code> .....	34, 36, 57		
<code>\scriptscriptstyle</code> .....			
	..... 33, 70, 76, 91, 98, 159, 378		
<code>\scriptstyle</code> .....	27,		
	36, 48, 49, 51, 54, 58, 75, 97, 156, 377		
<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> .....	92		
<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> .....	40		
<code>\seq_map_inline:Nn</code> .....	25, 35, 65, 126		
<code>\seq_new:N</code> .....	87–91, 100, 101		
<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> .....	19, 68, 108		
<code>\setboxz@h</code> .....	105		
<code>\SetMathAlphabet</code> .....	22,		
89–91, 95–97, 101, 102, 106, 107, 117			
<code>\SetMathAlphabet@</code> .....	117		
<code>\setmathfont</code> .....	3, 46, 107, 114		
<code>\setmathfontface</code> .....	7		
<code>\setmathrm</code> .....	24		
<code>\setoperatorfont</code> .....	12		
<code>\SetSymbolFont</code> .....			
... 116, 183, 187, 217, 222, 242, 247			
<code>\SetSymbolFont@</code> .....	116		
<code>\sfdefault</code> .....	71, 110, 115		
<code>\simeq</code> .....	172		
<code>\skip_set:Nn</code> .....	46		
<code>\slash</code> .....	125, 126		
<code>\smallint</code> .....	104		
<code>\smbllkcircle</code> .....	85		
<code>\smwhtdiamond</code> .....	88		
<code>\sp</code> .....	8		
<code>\space</code> .....	42, 136		
<code>\sqint</code> .....	360		
<code>\sqrt</code> .....	363		
<code>\sqrtsign</code> .....	28, 105, 147		
<code>\std@equal</code> .....	7, 12		
<code>\std@minus</code> .....	6, 11		
<code>\sterling</code> .....	87		
<code>\str_if_eq:eeT</code> .....	185, 220, 245		
<code>\str_if_eq:eeTF</code> .....	90		
<code>\string</code> .....	66–72		
<code>\subarray</code> .....	22		
<code>\sumint</code> .....	358		
<code>\sym</code> .....	72		
<code>\symit</code> .....	16		
<code>\symrm</code> .....	87		
<code>\symup</code> .....	15, 87		
<code>\sys_if_engine luatex:T</code> .....	52		
<code>\sys_if_engine pdftex:T</code> .....	54		
<code>\sys_if_engine xetex:T</code> .....	53		
		<b>T</b>	
<code>\tex_fam:D</code> .....	79		
<code>\tex_global:D</code> .....	26, 63, 67		
<code>\tex_iffontchar:D</code> .....	18		
<code>\tex_let:D</code> .....	26		
<code>\tex_mathcode:D</code> .....	63, 67		
<code>\textfont</code> .....	16, 19, 22, 25, 47, 48, 52		
<code>\textstyle</code> .....	21, 51, 74, 96, 104, 376		
<code>\tfrac</code> .....	5, 53, 189, 222		
<code>\the</code> .....	83, 383–427		
<code>\theta</code> .....	116		
<code>\thinmuskip</code> .....	105		
<code>\thr@@@</code> .....	31		
<code>\title</code> .....	30		
<code>\tl_case:Nn</code> .....	25, 38		
<code>\tl_clear:N</code> .....	8, 61, 62		
<code>\tl_const:cn</code> .....	101, 102		
<code>\tl_const:Nn</code> .....	3, 33–38		
<code>\tl_gclear_new:c</code> .....	22		
<code>\tl_gput_right:cx</code> .....	272		
<code>\tl_gset:Nn</code> .....	14, 150–152, 181–183		
<code>\tl_gset:Nx</code> .....	8, 143, 373		
<code>\tl_if_empty:NT</code> .....	16, 75, 133, 162		
<code>\tl_if_empty:nTF</code> .....	153, 155, 157, 168, 170		
<code>\tl_if_eq:nnF</code> .....	11		
<code>\tl_if_eq:nnT</code> .....	59		
<code>\tl_if_eq:nnTF</code> .....	120, 147		
<code>\tl_if_eq:onF</code> .....	109–111		
<code>\tl_if_eq:onT</code> .....	62, 71, 80, 114–116		
<code>\tl_if_exist:cF</code> .....	100		
<code>\tl_if_exist:NF</code> .....	32		
<code>\tl_if_in:NnT</code> .....	64, 67, 84		
<code>\tl_if_in:nnT</code> .....	17		
<code>\tl_if_in:NnTF</code> .....	89		
<code>\tl_if_in:VnTF</code> .....	106		
<code>\tl_if_single_p:n</code> .....	103		
<code>\tl_map_inline:nn</code> .....	108		
<code>\tl_new:N</code> .....	32, 39–72, 74		
<code>\tl_put_left:Nn</code> .....	31		
<code>\tl_put_right:Nn</code> .....	19, 32		
<code>\tl_put_right:NV</code> .....	46		
<code>\tl_remove_all:Nn</code> .....	23, 57, 71, 72		
<code>\tl_remove_once:Nn</code> .....	4, 125		
<code>\tl_set:cx</code> .....	14–16		
<code>\tl_set:Nn</code> .....	6, 7, 9, 12, 18, 20, 50, 52, 53,		
55, 56, 60, 87, 94, 102, 163, 355, 363			
<code>\tl_set:No</code> .....	37, 67, 69		
<code>\tl_set:Nx</code> .....	41, 70, 87, 93,		
94, 98, 99, 112, 114, 132, 135, 161, 164			





<code>\Uroot</code> .....	47, 85, 125	<code>\vee</code> .....	92
<code>\use:c</code> ..	8, 59, 102, 139, 144, 171, 195, 201	<code>\version@elt</code> .....	113
<code>\use:n</code> ...	63, 69, 75, 81, 138, 166, 172, 178	<code>\version@list</code> .....	113
<code>\use:x</code> .....	5	<code>\Vert</code> .....	122
<code>\use@mathgroup</code> .....	<u>32</u>	<code>\vrule</code> .....	96, 104
<code>\use_i:nnn</code> .....	37, 67	<code>\vysmwhtcircle</code> .....	84
<code>\use_ii:nnn</code> .....	38, 68		
<code>\use_iii:nnn</code> .....	69	<b>W</b>	
<code>\use_none:n</code> .....	3, 27, 31, 44	<code>\wd</code> .....	109
<code>\use_none:nnn</code> .....	136	<code>\wedge</code> .....	91
<code>\use_none_delimit_by_q_nil:w</code>	123, 132		
<code>\usv_set:nnn</code> .....	2–245, 248–419	<b>X</b>	
<code>\usv_set:nnn,\@@_to_usv:nn</code> .....	<u>101</u>	<code>\XeTeXradical</code> .....	72
<code>\Uunderbracket</code> .....	61, 109		
		<b>Y</b>	
<b>V</b>		<code>\yen</code> .....	86
<code>\varnothing</code> .....	89		
<code>\varointclockwise</code> .....	358	<b>Z</b>	
<code>\vcenter</code> .....	24, 41	<code>\Z</code> .....	151
		<code>\z@</code> ....	19, 20, 45, 62–64, 96, 104, 106, 107