

# The **boustrophedon** package

Aris Chatzichristos, PhD

Version 1.1.1, 2026-05-20

## Abstract

**boustrophedon** typesets text in boustrophedon style, alternating writing direction line by line. It supports generic mirrored text, Archaic and Classical Greek glyph rendering, automatic wrapping, explicit line breaks, and inline bracketed spans.

## 1 Overview

This package typesets text in *boustrophedon* style, alternating the writing direction on each line.

Have you ever wondered why most Western languages are written left-to-right, while many Middle Eastern languages are written right-to-left? Both conventions may seem arbitrary today—but this was not always the case.

In one of the earliest fully developed alphabets, the Archaic Greek alphabet (from *alpha* and *beta*, hence “alphabet”), writing was often performed *boustrophedon*, meaning “as the ox turns in ploughing.” Text was written left-to-right on one line, then right-to-left on the next, with both the direction of the text *and the letterforms themselves* mirrored accordingly.

This method creates a continuous flow of reading, as the reader does not need to jump back to the start of each new line, but instead follows the text in a natural back-and-forth motion.

Reading takes some getting used to, but after a while it becomes surprisingly natural. One may also observe that:

- Capital letters (both in the Latin alphabet and especially in the Greek alphabet) are particularly well-suited to this type of writing, and are much easier to read than lowercase letters (which were developed later in both systems).
- For example, pairs such as *b* vs *d*, or comparable look-alikes in polytonic Greek, can make reading more difficult, as the reader must keep mental track of the text direction.
- In contrast, capital Greek letters—already in use during the boustrophedon period—have the appropriate symmetry to be read in both directions without ambiguity.

This package brings an ancient writing paradigm into modern digital typography.

## 2 Installation

To generate the package file from this documented source, run:

```
latex boustrophedon.ins
```

This produces **boustrophedon.sty**. Place it beside your document or install it in a TeX tree searched by your distribution.

## 3 Requirements

XeLaTeX or LuaLaTeX is recommended, especially for Greek Unicode input. The package requires **xparse**, **expl3**, **graphicx**, **iftex**, **greek6cbc**, and **greek4cbc**.

## 4 Usage

The main command is:

```
\boustrophedon[<keys>]{<text>}
```

Important keys are:

- Type=Any, Type=Archaic, or Type=Classical
- start=LTR or start=RTL
- resetDirection=Page or resetDirection=Paragraph
- LineWidth=Auto or an explicit dimension
- ObscureLetters=On or ObscureLetters=Off
- inLine=True or inLine=False

Inline mode keeps surrounding text normal and transforms only bracketed spans:

```
\boustrophedon[inLine=True]{Normal text with [THIS PART] mirrored inline.}
\boustrophedon[Type=Classical,inLine=True]{A sentence with [DHMOS ATHNAIWN].}
```

The start/stop form is also available:

```
\startBoustrophedon[Type=Archaic]
ANDRA MOI ENNEPE MOUSA POLYTROPON ...
\stopBoustrophedon
```

## 5 Examples

See `examples/boustrophedon_template.tex` for a minimal starter and `examples/boustrophedon_demos.tex` for the full demonstration and regression document.

## 6 License and maintenance

This work is distributed under the LaTeX Project Public License, version 1.3c or later. The work has LPPL maintenance status “maintained”. The Current Maintainer is Aris Chatzichristos.

## 7 Implementation

```
1 <*package>
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{boustrophedon}[2026/05/20 v1.1.1 Documented source distribution]
4
5 \RequirePackage{xparse}
6 \RequirePackage{expl3}
7 \RequirePackage{graphicx}
8 \RequirePackage{greek6cbc}
9 \RequirePackage{greek4cbc}
10 \RequirePackage{iftex}
11
12 \ExplSyntaxOn
13
14 % -----
15 % Options
16 % -----
17 \bool_new:N \g_boustro_start_ltr_bool
```

```

18 \bool_new:N \g_boustro_obscure_letters_bool
19 \bool_new:N \g_boustro_reset_page_bool
20 \bool_new:N \g_boustro_inline_bool
21 \int_new:N \g_boustro_line_int
22 \tl_new:N \g_boustro_type_tl
23 \dim_new:N \g_boustro_linewidth_dim
24
25 \bool_set_true:N \g_boustro_start_ltr_bool
26 \bool_set_true:N \g_boustro_obscure_letters_bool
27 \bool_set_true:N \g_boustro_reset_page_bool
28 \bool_set_false:N \g_boustro_inline_bool
29 \int_gzero:N \g_boustro_line_int
30 \tl_set:Nn \g_boustro_type_tl {any}
31 \dim_set:Nn \g_boustro_linewidth_dim { 0pt }
32
33 \cs_new_protected:Npn \boustro_set_linewidth:n #1
34 {
35   \tl_set:Nn \l_tmpb_tl {#1}
36   \tl_trim_spaces:N \l_tmpb_tl
37   \tl_set:Nv \l_tmpa_tl \l_tmpb_tl
38   \tl_set:Nx \l_tmpa_tl { \tl_lower_case:n { \l_tmpa_tl } }
39   \tl_if_blank:VTF \l_tmpa_tl
40   { \dim_set:Nn \g_boustro_linewidth_dim { 0pt } }
41   {
42     \str_if_eq:VnTF \l_tmpa_tl { auto }
43     { \dim_set:Nn \g_boustro_linewidth_dim { 0pt } }
44     {
45       \dim_set:Nn \g_boustro_linewidth_dim { \l_tmpb_tl }
46     }
47   }
48 }
49
50 \keys_define:nn { boustro }
51 {
52   start .choice:,
53   start / LTR .code:n = { \bool_set_true:N \g_boustro_start_ltr_bool },
54   start / RTL .code:n = { \bool_set_false:N \g_boustro_start_ltr_bool },
55   Type .choice:,
56   Type / Any .code:n = { \tl_set:Nn \g_boustro_type_tl {any} },
57   Type / Archaic .code:n = { \tl_set:Nn \g_boustro_type_tl {archaic} },
58   Type / Classical .code:n = { \tl_set:Nn \g_boustro_type_tl {classical} },
59   LineWidth .code:n = { \boustro_set_linewidth:n {#1} },
60   ObscureLetters .choice:,
61   ObscureLetters / On .code:n = { \bool_set_true:N \g_boustro_obscure_letters_bool },
62   ObscureLetters / Off .code:n = { \bool_set_false:N \g_boustro_obscure_letters_bool },
63   % Backwards-compatible alias (deprecated):
64   digammaAndKoppa .choice:,
65   digammaAndKoppa / On .code:n = { \bool_set_true:N \g_boustro_obscure_letters_bool },
66   digammaAndKoppa / Off .code:n = { \bool_set_false:N \g_boustro_obscure_letters_bool },
67   resetDirection .choice:,
68   resetDirection / Page .code:n = { \bool_set_true:N \g_boustro_reset_page_bool },
69   resetDirection / Paragraph .code:n = { \bool_set_false:N \g_boustro_reset_page_bool },
70   % Backwards-compatible alias (deprecated):
71   reset .choice:,
72   reset / page .code:n = { \bool_set_true:N \g_boustro_reset_page_bool },
73   reset / paragraph .code:n = { \bool_set_false:N \g_boustro_reset_page_bool },
74   inline .choice:,
75   inline / True .code:n = { \bool_set_true:N \g_boustro_inline_bool },
76   inline / False .code:n = { \bool_set_false:N \g_boustro_inline_bool },
77   inline / true .code:n = { \bool_set_true:N \g_boustro_inline_bool },
78   inline / false .code:n = { \bool_set_false:N \g_boustro_inline_bool },
79   inline .choice:,
80   inline / True .code:n = { \bool_set_true:N \g_boustro_inline_bool },

```

```

81 inline / False .code:n = { \bool_set_false:N \g_boustro_inline_bool },
82 inline / true .code:n = { \bool_set_true:N \g_boustro_inline_bool },
83 inline / false .code:n = { \bool_set_false:N \g_boustro_inline_bool },
84 }
85 \ProcessKeyOptions [ boustro ]
86
87 \AddToHook{shipout/after}
88 {
89   \bool_if:NT \g_boustro_reset_page_bool
90     { \int_gzero:N \g_boustro_line_int }
91 }
92
93 % -----
94 % Variables
95 % -----
96 \seq_new:N \__boustro_lines_seq
97 \seq_new:N \__boustro_symbols_seq
98 \seq_new:N \__boustro_tmp_seq
99 \seq_new:N \__boustro_words_seq
100 \seq_new:N \__boustro_pars_seq
101 \bool_new:N \__boustro_input_has_greek_bool
102 \tl_new:N \__boustro_norm_tl
103 \tl_new:N \__boustro_scan_tl
104 \tl_new:N \__boustro_scan_tail_tl
105 \tl_new:N \__boustro_head_tl
106 \tl_new:N \__boustro_next_tl
107 \tl_new:N \__boustro_combined_tl
108 \tl_new:N \__boustro_current_line_tl
109 \tl_new:N \__boustro_candidate_tl
110 \tl_new:N \__boustro_words_tl
111 \tl_new:N \__boustro_input_tl
112 \tl_new:N \__boustro_inline_mode_tl
113 \box_new:N \__boustro_measure_box
114 \dim_new:N \__boustro_width_dim
115 \dim_new:N \__boustro_target_dim
116
117 \cs_new_protected:Npn \boustro_set_target_width:n #1
118 {
119   \dim_compare:nNnTF { \g_boustro_linewidth_dim } > { 0pt }
120     { \dim_set_eq:NN \__boustro_target_dim \g_boustro_linewidth_dim }
121     {
122       \str_case:nnF {#1}
123       {
124         {any}{\dim_set:Nn \__boustro_target_dim { .95\linewidth }}
125         {archaic}{\dim_set:Nn \__boustro_target_dim { .95\linewidth }}
126         {classical}{\dim_set:Nn \__boustro_target_dim { .95\linewidth }}
127       }
128       { \dim_set:Nn \__boustro_target_dim { .95\linewidth } }
129     }
130   \dim_compare:nNnT { \__boustro_target_dim } > { \linewidth }
131     { \dim_set:Nn \__boustro_target_dim { \linewidth } }
132 }
133
134 \cs_new_protected:Npn \boustro_maybe_reset:
135 {
136   \bool_if:NF \g_boustro_reset_page_bool
137     { \int_gzero:N \g_boustro_line_int }
138 }
139
140 \cs_new:Npn \boustro_line_is_reverse:
141 {
142   \bool_if:NTF \g_boustro_start_ltr_bool
143     { \int_if_even_p:n { \g_boustro_line_int } }

```



```

207 \regex_replace_all:nnN { Γ } { G } \l__boustro_norm_tl
208 \regex_replace_all:nnN { δ } { d } \l__boustro_norm_tl
209 \regex_replace_all:nnN { Δ } { D } \l__boustro_norm_tl
210 \regex_replace_all:nnN { ε } { e } \l__boustro_norm_tl
211 \regex_replace_all:nnN { Ε } { E } \l__boustro_norm_tl
212 \regex_replace_all:nnN { ζ } { z } \l__boustro_norm_tl
213 \regex_replace_all:nnN { Ζ } { Z } \l__boustro_norm_tl
214 \regex_replace_all:nnN { η } { h } \l__boustro_norm_tl
215 \regex_replace_all:nnN { Η } { H } \l__boustro_norm_tl
216 \regex_replace_all:nnN { Θ } { 1 } \l__boustro_norm_tl
217 \regex_replace_all:nnN { Θ } { 1 } \l__boustro_norm_tl
218 \regex_replace_all:nnN { ι|ϊ|ί } { i } \l__boustro_norm_tl
219 \regex_replace_all:nnN { Ι|ϊ|ί } { I } \l__boustro_norm_tl
220 \regex_replace_all:nnN { κ } { k } \l__boustro_norm_tl
221 \regex_replace_all:nnN { Κ } { K } \l__boustro_norm_tl
222 \regex_replace_all:nnN { λ } { l } \l__boustro_norm_tl
223 \regex_replace_all:nnN { Λ } { L } \l__boustro_norm_tl
224 \regex_replace_all:nnN { μ } { m } \l__boustro_norm_tl
225 \regex_replace_all:nnN { Μ } { M } \l__boustro_norm_tl
226 \regex_replace_all:nnN { ν } { n } \l__boustro_norm_tl
227 \regex_replace_all:nnN { Ν } { N } \l__boustro_norm_tl
228 \regex_replace_all:nnN { ξ } { 5 } \l__boustro_norm_tl
229 \regex_replace_all:nnN { Ξ } { 5 } \l__boustro_norm_tl
230 \regex_replace_all:nnN { ο } { o } \l__boustro_norm_tl
231 \regex_replace_all:nnN { Ο } { O } \l__boustro_norm_tl
232 \regex_replace_all:nnN { π } { p } \l__boustro_norm_tl
233 \regex_replace_all:nnN { Π } { P } \l__boustro_norm_tl
234 \regex_replace_all:nnN { ρ } { r } \l__boustro_norm_tl
235 \regex_replace_all:nnN { Ρ } { R } \l__boustro_norm_tl
236 \regex_replace_all:nnN { σ|ς } { s } \l__boustro_norm_tl
237 \regex_replace_all:nnN { Σ } { S } \l__boustro_norm_tl
238 \regex_replace_all:nnN { τ } { t } \l__boustro_norm_tl
239 \regex_replace_all:nnN { Τ } { T } \l__boustro_norm_tl
240 \regex_replace_all:nnN { υ|ϋ|ϋ } { u } \l__boustro_norm_tl
241 \regex_replace_all:nnN { Υ|Υ } { U } \l__boustro_norm_tl
242 \regex_replace_all:nnN { φ } { 2 } \l__boustro_norm_tl
243 \regex_replace_all:nnN { Φ } { 2 } \l__boustro_norm_tl
244 \regex_replace_all:nnN { χ } { 3 } \l__boustro_norm_tl
245 \regex_replace_all:nnN { Χ } { 3 } \l__boustro_norm_tl
246 \regex_replace_all:nnN { ψ } { 4 } \l__boustro_norm_tl
247 \regex_replace_all:nnN { Ψ } { 4 } \l__boustro_norm_tl
248 \regex_replace_all:nnN { ω } { 6 } \l__boustro_norm_tl
249 \regex_replace_all:nnN { Ω } { 6 } \l__boustro_norm_tl
250 }
251
252 \cs_new_protected:Npn \boustro_normalize_input:n #1
253 {
254   \tl_set:Nn \l__boustro_norm_tl {#1}
255   \regex_match:nnTF { [\x{0370}-\x{03FF}]\x{1F00}-\x{1FFF} } {#1}
256   { \bool_set_true:N \l__boustro_input_has_greek_bool }
257   { \bool_set_false:N \l__boustro_input_has_greek_bool }
258   \boustro_normalize_unicode_greek:
259
260   \bool_if:NF \l__boustro_input_has_greek_bool
261   {
262     \regex_replace_all:nnN { J } { I } \l__boustro_norm_tl
263     \regex_replace_all:nnN { j } { i } \l__boustro_norm_tl
264     \regex_replace_all:nnN { Ô|ô|Ô|ô } { 6 } \l__boustro_norm_tl
265     \bool_if:NT \g_boustro_obscure_letters_bool
266     {
267       \regex_replace_all:nnN { TH|Th|tH|th } { 1 } \l__boustro_norm_tl
268       \regex_replace_all:nnN { PH|Ph|pH|ph } { 2 } \l__boustro_norm_tl
269       \regex_replace_all:nnN { CH|Ch|cH|ch|KH|Kh|kH|kh } { 3 } \l__boustro_norm_tl

```

```

270 \regex_replace_all:nnN { PS|Ps|pS|ps } { 4 } \__boustro_norm_tl
271 \regex_replace_all:nnN { KS|Ks|kS|ks } { 5 } \__boustro_norm_tl
272 \regex_replace_all:nnN { KA } { 7A } \__boustro_norm_tl
273 \regex_replace_all:nnN { Ka } { 7a } \__boustro_norm_tl
274 \regex_replace_all:nnN { kA } { 7A } \__boustro_norm_tl
275 \regex_replace_all:nnN { ka } { 7a } \__boustro_norm_tl
276 \regex_replace_all:nnN { KO } { 7O } \__boustro_norm_tl
277 \regex_replace_all:nnN { Ko } { 7o } \__boustro_norm_tl
278 \regex_replace_all:nnN { kO } { 7O } \__boustro_norm_tl
279 \regex_replace_all:nnN { ko } { 7o } \__boustro_norm_tl
280 \regex_replace_all:nnN { W } { 8 } \__boustro_norm_tl
281 \regex_replace_all:nnN { w } { 8 } \__boustro_norm_tl
282 }
283 }
284
285 \regex_replace_all:nnN { [.,:;!]? } { ~ } \__boustro_norm_tl
286 \regex_replace_all:nnN { [-()\\[]"'] } { ~ } \__boustro_norm_tl
287 \regex_replace_all:nnN { [^A-Za-z0-9~ ] } { } \__boustro_norm_tl
288 % Normalize any remaining whitespace to a visible/measurable separator.
289 % Using '~' lets us map it to an explicit '\space' during rendering so it
290 % isn't lost inside macro expansions (important for width-based wrapping).
291 \regex_replace_all:nnN { \s+ } { ~ } \__boustro_norm_tl
292 \regex_replace_all:nnN { ~+ } { ~ } \__boustro_norm_tl
293 }
294
295 \cs_new_protected:Npn \__boustro_line_to_seq:n #1
296 {
297   \seq_clear:N \__boustro_symbols_seq
298   \boustro_normalize_input:n {#1}
299   % Treat certain digraph encodings as atomic tokens (e.g. 7A/7O for koppa+vowel)
300   % so reversal doesn't split them into separate characters.
301   \tl_set_eq:NN \__boustro_scan_tl \__boustro_norm_tl
302   \boustro_scan_norm_into_symbols:
303 }
304
305 \cs_new_protected:Npn \__boustro_scan_norm_into_symbols:
306 {
307   \tl_if_empty:NTF \__boustro_scan_tl
308   { }
309   {
310     \tl_set:Nx \__boustro_head_tl { \tl_head:N \__boustro_scan_tl }
311     \tl_set:Nx \__boustro_scan_tail_tl { \tl_tail:N \__boustro_scan_tl }
312
313     \tl_if_empty:NTF \__boustro_scan_tail_tl
314     {
315       \seq_put_right:NV \__boustro_symbols_seq \__boustro_head_tl
316       \tl_clear:N \__boustro_scan_tl
317     }
318     {
319       \tl_set:Nx \__boustro_next_tl { \tl_head:N \__boustro_scan_tail_tl }
320       \tl_if_eq:NnTF \__boustro_head_tl { 7 }
321       {
322         \str_case:nnF { \__boustro_next_tl }
323         {
324           {A}{ \tl_set:Nn \__boustro_combined_tl { 7A } }
325           {a}{ \tl_set:Nn \__boustro_combined_tl { 7a } }
326           {O}{ \tl_set:Nn \__boustro_combined_tl { 7O } }
327           {o}{ \tl_set:Nn \__boustro_combined_tl { 7o } }
328         }
329         { \tl_clear:N \__boustro_combined_tl }
330
331       \tl_if_blank:NTF \__boustro_combined_tl
332       {

```

```

333     \seq_put_right:NV \l__boustro_symbols_seq \l__boustro_head_tl
334     \tl_set_eq:NN \l__boustro_scan_tl \l__boustro_scan_tail_tl
335   }
336   {
337     \seq_put_right:NV \l__boustro_symbols_seq \l__boustro_combined_tl
338     \tl_set:Nx \l__boustro_scan_tl { \tl_tail:N \l__boustro_scan_tail_tl }
339   }
340 }
341 {
342   \seq_put_right:NV \l__boustro_symbols_seq \l__boustro_head_tl
343   \tl_set_eq:NN \l__boustro_scan_tl \l__boustro_scan_tail_tl
344 }
345 }
346 \boustro_scan_norm_into_symbols:
347 }
348 }
349
350 % -----
351 % Glyph mapping helpers
352 % -----
353 \cs_new:Npn \boustro_archaic_map:n #1
354 {
355   \str_case:nnF {#1}
356   {
357     {~}{\space}
358     {A}{\Aalpha}{a}{\Aalpha}
359     {B}{\Abeta}{b}{\Abeta}
360     {G}{\Agamma}{g}{\Agamma}
361     {D}{\Adelta}{d}{\Adelta}
362     {E}{\Aepsilon}{e}{\Aepsilon}
363     {Z}{\Azeta}{z}{\Azeta}
364     {H}{\Aeta}{h}{\Aeta}
365     {1}{\Atheta}
366     {I}{\Aiota}{i}{\Aiota}
367     {K}{\Akappa}{k}{\Akappa}
368     {7}{\Akoppa}
369     {7A}{\Akoppa\Aalpha}{7a}{\Akoppa\Aalpha}
370     {7O}{\Akoppa\Aomicron}{7o}{\Akoppa\Aomicron}
371     {L}{\Alambda}{l}{\Alambda}
372     {M}{\Amu}{m}{\Amu}
373     {N}{\Anu}{n}{\Anu}
374     {X}{\Axi}{x}{\Axi}{5}{\Axi}
375     {O}{\Aomicron}{o}{\Aomicron}
376     {6}{\Aomega}
377     {P}{\Api}{p}{\Api}
378     {R}{\Arho}{r}{\Arho}
379     {S}{\Asigma}{s}{\Asigma}
380     {T}{\Atau}{t}{\Atau}
381     {U}{\Aupsilon}{u}{\Aupsilon}
382     {Y}{\Aupsilon}{y}{\Aupsilon}
383     {3}{\Achi}
384     {2}{\Aphi}
385     {4}{\Apsi}
386     {8}{\Adigamma}
387   }
388   {#1}
389 }
390
391 \cs_new:Npn \boustro_classical_normal_map:n #1
392 {
393   \str_case:nnF {#1}
394   {
395     {~}{\space}

```



```

396 {A}{\Aalpha}{a}{\Aalpha}
397 {B}{\Abeta}{b}{\Abeta}
398 {G}{\Agamma}{g}{\Agamma}
399 {D}{\Adelta}{d}{\Adelta}
400 {E}{\Aepsilon}{e}{\Aepsilon}
401 {Z}{\Azeta}{z}{\Azeta}
402 {H}{\Aeta}{h}{\Aeta}
403 {1}{\Atheta}
404 {I}{\Aiota}{i}{\Aiota}
405 {K}{\Akappa}{k}{\Akappa}
406 {L}{\Alambda}{l}{\Alambda}
407 {M}{\Amu}{m}{\Amu}
408 {N}{\Anu}{n}{\Anu}
409 {X}{\Axi}{x}{\Axi}{5}{\Axi}
410 {O}{\Aomicron}{o}{\Aomicron}
411 {6}{\Aomega}
412 {P}{\Api}{p}{\Api}
413 {R}{\Arho}{r}{\Arho}
414 {S}{\Asigma}{s}{\Asigma}
415 {T}{\Atau}{t}{\Atau}
416 {U}{\Aupsilon}{u}{\Aupsilon}
417 {Y}{\Aupsilon}{y}{\Aupsilon}
418 {3}{\Achi}
419 {2}{\Aphi}
420 {4}{\Apsi}
421 }
422 {
423 \str_if_eq:nnTF {#1}{7A}
424 { \textgvibc{\Akoppa}\Aalpha }
425 {
426 \str_if_eq:nnTF {#1}{7a}
427 { \textgvibc{\Akoppa}\Aalpha }
428 {
429 \str_if_eq:nnTF {#1}{7O}
430 { \textgvibc{\Akoppa}\Aomicron }
431 {
432 \str_if_eq:nnTF {#1}{7o}
433 { \textgvibc{\Akoppa}\Aomicron }
434 {
435 \str_if_eq:nnTF {#1}{7}
436 { \textgvibc{\Akoppa} }
437 {
438 \str_if_eq:nnTF {#1}{8}
439 { \textgvibc{\Adigamma} }
440 {#1}
441 }
442 }
443 }
444 }
445 }
446 }
447 }
448
449 \cs_new:Npn \boustro_classical_reverse_map:n #1
450 {
451 \str_case:nnF {#1}
452 {
453 {~}{\space}
454 {A}{\ARalpha}{a}{\ARalpha}
455 {B}{\ARbeta}{b}{\ARbeta}
456 {G}{\ARgamma}{g}{\ARgamma}
457 {D}{\ARdelta}{d}{\ARdelta}
458 {E}{\AREpsilon}{e}{\AREpsilon}

```

```

459 {Z}{\ARzeta}{z}{\ARzeta}
460 {H}{\AReta}{h}{\AReta}
461 {1}{\ARtheta}
462 {I}{\ARiota}{i}{\ARiota}
463 {K}{\ARKappa}{k}{\ARKappa}
464 {L}{\ARlambda}{l}{\ARlambda}
465 {M}{\ARMu}{m}{\ARMu}
466 {N}{\ARnu}{n}{\ARnu}
467 {X}{\ARxi}{x}{\ARxi}{5}{\ARxi}
468 {O}{\ARomicron}{o}{\ARomicron}
469 {6}{\ARomega}
470 {P}{\ARpi}{p}{\ARpi}
471 {R}{\ARRho}{r}{\ARRho}
472 {S}{\ARsigma}{s}{\ARsigma}
473 {T}{\ARTau}{t}{\ARTau}
474 {U}{\ARupsilon}{u}{\ARupsilon}
475 {Y}{\ARupsilon}{y}{\ARupsilon}
476 {3}{\ARchi}
477 {2}{\ARphi}
478 {4}{\ARpsi}
479 }
480 {
481   \str_if_eq:nnTF {#1}{7A}
482   { \textgvibc{\Akoppa}\ARalpha }
483   {
484     \str_if_eq:nnTF {#1}{7a}
485     { \textgvibc{\Akoppa}\ARalpha }
486     {
487       \str_if_eq:nnTF {#1}{7O}
488       { \textgvibc{\Akoppa}\ARomicron }
489       {
490         \str_if_eq:nnTF {#1}{7o}
491         { \textgvibc{\Akoppa}\ARomicron }
492         {
493           \str_if_eq:nnTF {#1}{7}
494           { \textgvibc{\Akoppa} }
495           {
496             \str_if_eq:nnTF {#1}{8}
497             { \textgvibc{\Adigamma} }
498             {#1}
499           }
500         }
501       }
502     }
503   }
504 }
505 }
506
507 \cs_new_protected:Npn \boustro_render_archaic_forward:
508 { \textgvibc{\seq_map_inline:Nn \l__boustro_symbols_seq { \boustro_archaic_map:n{##1} }}} }
509 \cs_new_protected:Npn \boustro_render_archaic_reverse:
510 {
511   \seq_set_eq:NN \l__boustro_tmp_seq \l__boustro_symbols_seq
512   \seq_reverse:N \l__boustro_tmp_seq
513   {
514     \textgvibc
515     {
516       \seq_map_inline:Nn \l__boustro_tmp_seq
517       {
518         \str_if_eq:nnTF {##1} {~}
519         { \space }
520         { \reflectbox { \boustro_archaic_map:n {##1} } }
521       }

```

```

522     }
523   }
524 }
525 \cs_new_protected:Npn \boustro_render_classical_forward:
526 { {\textgivbc{\seq_map_inline:Nn \l__boustro_symbols_seq {\boustro_classical_normal_map:n{##1}}}} }
527 \cs_new_protected:Npn \boustro_render_classical_reverse:
528 {
529   \seq_set_eq:NN \l__boustro_tmp_seq \l__boustro_symbols_seq
530   \seq_reverse:N \l__boustro_tmp_seq
531   {
532     \textgivbc
533     {
534       \seq_map_inline:Nn \l__boustro_tmp_seq
535       {
536         \str_if_eq:nnTF {##1} {~}
537         { \space }
538         { \reflectbox { \boustro_classical_normal_map:n {##1} } }
539       }
540     }
541   }
542 }
543
544 % -----
545 % Width measurement and wrapping
546 % -----
547 \cs_new_protected:Npn \boustro_measure_any:nN #1#2
548 {
549   \hbox_set:Nn \l__boustro_measure_box {#1}
550   \dim_set:Nn #2 { \box_wd:N \l__boustro_measure_box }
551 }
552
553 \cs_new_protected:Npn \boustro_measure_archaic:nN #1#2
554 {
555   \boustro_line_to_seq:n {#1}
556   \hbox_set:Nn \l__boustro_measure_box { \boustro_render_archaic_forward: }
557   \dim_set:Nn #2 { \box_wd:N \l__boustro_measure_box }
558 }
559
560 \cs_new_protected:Npn \boustro_measure_classical:nN #1#2
561 {
562   \boustro_line_to_seq:n {#1}
563   \hbox_set:Nn \l__boustro_measure_box { \boustro_render_classical_forward: }
564   \dim_set:Nn #2 { \box_wd:N \l__boustro_measure_box }
565 }
566 \cs_generate_variant:Nn \boustro_measure_any:nN { VN }
567 \cs_generate_variant:Nn \boustro_measure_archaic:nN { VN }
568 \cs_generate_variant:Nn \boustro_measure_classical:nN { VN }
569
570 \cs_new:Npn \boustro_map_string_archaic:n #1
571 {
572   \boustro_line_to_seq:n {#1}
573   \seq_map_inline:Nn \l__boustro_symbols_seq { \boustro_archaic_map:n{##1} }
574 }
575
576 \cs_new:Npn \boustro_map_string_classical:n #1
577 {
578   \boustro_line_to_seq:n {#1}
579   \seq_map_inline:Nn \l__boustro_symbols_seq { \boustro_classical_normal_map:n{##1} }
580 }
581
582 \cs_new_protected:Npn \boustro_wrap_text:nnN #1#2#3
583 {
584   \seq_clear:N #3

```

```

585 \boustro_set_target_width:n {#1}
586 \tl_set:Nx \l__boustro_words_tl { \tl_to_str:n {#2} }
587 \regex_replace_all:nnN { \s+ } { <BSPACE> } \l__boustro_words_tl
588 \seq_set_split:NnV \l__boustro_words_seq { <BSPACE> } \l__boustro_words_tl
589 \seq_remove_all:Nn \l__boustro_words_seq { }
590 \tl_clear:N \l__boustro_current_line_tl
591 \seq_map_inline:Nn \l__boustro_words_seq
592 {
593   \tl_if_blank:VTF \l__boustro_current_line_tl
594   { \tl_set:Nn \l__boustro_candidate_tl {##1} }
595   { \tl_set:Nc \l__boustro_candidate_tl { \l__boustro_current_line_tl \c_space_tl ##1 } }
596   \use:c { boustro_measure_#1:VN } \l__boustro_candidate_tl \l__boustro_width_dim
597   \dim_compare:nNnTF { \l__boustro_width_dim } > { \l__boustro_target_dim }
598   {
599     \tl_if_blank:VTF \l__boustro_current_line_tl
600     { \seq_put_right:Nn #3 {##1} }
601     {
602       \seq_put_right:NV #3 \l__boustro_current_line_tl
603       \tl_set:Nn \l__boustro_current_line_tl {##1}
604     }
605   }
606   { \tl_set:NV \l__boustro_current_line_tl \l__boustro_candidate_tl }
607 }
608 \tl_if_blank:VF \l__boustro_current_line_tl
609 { \seq_put_right:NV #3 \l__boustro_current_line_tl }
610 }
611
612 \cs_new_protected:Npn \boustro_wrap_epigraphic_text:nnN #1#2#3
613 {
614   \seq_clear:N #3
615   \boustro_normalize_input:n {#2}
616   \tl_set_eq:NN \l__boustro_words_tl \l__boustro_norm_tl
617   \regex_replace_all:nnN { \s+ } { <BSPACE> } \l__boustro_words_tl
618   \seq_set_split:NnV \l__boustro_words_seq { <BSPACE> } \l__boustro_words_tl
619   \seq_remove_all:Nn \l__boustro_words_seq { }
620   \boustro_set_target_width:n {#1}
621   \tl_clear:N \l__boustro_current_line_tl
622   \seq_map_inline:Nn \l__boustro_words_seq
623   {
624     \tl_if_blank:VTF \l__boustro_current_line_tl
625     { \tl_set:Nn \l__boustro_candidate_tl {##1} }
626     { \tl_set:Nc \l__boustro_candidate_tl { \l__boustro_current_line_tl \c_space_tl ##1 } }
627     \use:c { boustro_measure_#1:VN } \l__boustro_candidate_tl \l__boustro_width_dim
628     \dim_compare:nNnTF { \l__boustro_width_dim } > { \l__boustro_target_dim }
629     {
630       \tl_if_blank:VTF \l__boustro_current_line_tl
631       { \seq_put_right:Nn #3 {##1} }
632       {
633         \seq_put_right:NV #3 \l__boustro_current_line_tl
634         \tl_set:Nn \l__boustro_current_line_tl {##1}
635       }
636     }
637     { \tl_set:NV \l__boustro_current_line_tl \l__boustro_candidate_tl }
638   }
639   \tl_if_blank:VF \l__boustro_current_line_tl
640   { \seq_put_right:NV #3 \l__boustro_current_line_tl }
641 }
642
643 \cs_new_protected:Npn \boustro_prepare_lines:nn #1#2
644 {
645   \seq_clear:N \l__boustro_lines_seq
646   \tl_set:Nn \l__boustro_input_tl {#2}
647   \tl_if_in:NnTF \l__boustro_input_tl {\}

```

```

648 {
649   \boustro_split_explicit_lines:n {#2}
650   \seq_map_inline:Nn \l__boustro_pars_seq
651   {
652     \boustro_wrap_text:nnN {#1} {##1} \l__boustro_tmp_seq
653     \seq_map_inline:Nn \l__boustro_tmp_seq
654     { \seq_put_right:Nn \l__boustro_lines_seq {###1} }
655   }
656 }
657 {
658   \str_case:nnF {#1}
659   {
660     {any}{\boustro_wrap_text:nnN {any} {#2} \l__boustro_lines_seq}
661     {archaic}{\boustro_wrap_text:nnN {archaic} {#2} \l__boustro_lines_seq}
662     {classical}{\boustro_wrap_text:nnN {classical} {#2} \l__boustro_lines_seq}
663   }
664   { \boustro_wrap_text:nnN {#1} {#2} \l__boustro_lines_seq }
665 }
666 }
667
668 % -----
669 % Printing line by line
670 % -----
671 \cs_new_protected:Npn \boustro_print_any_line:nn #1#2
672 {
673   \par\noindent
674   \bool_if:nTF { \boustro_line_is_reverse_n:n {#1} }
675   { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][r]{\reflectbox{#2}}}\par }
676   { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][l]{#2}}\par }
677 }
678
679 \cs_new_protected:Npn \boustro_print_archaic_line:nn #1#2
680 {
681   \par\noindent
682   \boustro_line_to_seq:n {#2}
683   \bool_if:nTF { \boustro_line_is_reverse_n:n {#1} }
684   { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][r]{\boustro_render_archaic_reverse:}}\par }
685   { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][l]{\boustro_render_archaic_forward:}}\par }
686 }
687
688 \cs_new_protected:Npn \boustro_print_classical_line:nn #1#2
689 {
690   \par\noindent
691   \boustro_line_to_seq:n {#2}
692   \bool_if:nTF { \boustro_line_is_reverse_n:n {#1} }
693   {
694     \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][r]{\boustro_render_classical_reverse:}}\par
695   }
696   {
697     \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][l]{\boustro_render_classical_forward:}}\par
698   }
699 }
700
701 % -----
702 % Public typesetter (mode, text)
703 % -----
704 \cs_new_protected:Npn \boustro_typeset_block:nn #1#2
705 {
706   \boustro_maybe_reset:
707   \boustro_set_target_width:n {#1}
708   \boustro_prepare_lines:nn {#1} {#2}
709   \int_set:Nn \l__tmpa_int { \g_boustro_line_int }
710   \str_case:nnF {#1}

```

```

711 {
712 {any}
713 {
714 \seq_map_indexed_inline:Nn \__boustro_lines_seq
715 { \boustro_print_any_line:nn { \int_eval:n { \l_tmpa_int + ##1 } } {##2} }
716 }
717 {archaic}
718 {
719 \seq_map_indexed_inline:Nn \__boustro_lines_seq
720 { \boustro_print_archaic_line:nn { \int_eval:n { \l_tmpa_int + ##1 } } {##2} }
721 }
722 {classical}
723 {
724 \seq_map_indexed_inline:Nn \__boustro_lines_seq
725 { \boustro_print_classical_line:nn { \int_eval:n { \l_tmpa_int + ##1 } } {##2} }
726 }
727 }
728 \int_gadd:Nn \g_boustro_line_int { \seq_count:N \__boustro_lines_seq }
729 \par
730 }
731
732 % -----
733 % Inline markup: normal text outside [ ... ], boustrophedon only inside.
734 % The square brackets are markup delimiters and are not printed.
735 % -----
736 \cs_new_protected:Npn \boustro_render_inline:nn #1#2
737 {
738 \group_begin:
739 \str_case:VnF \__boustro_inline_mode_tl
740 {
741 {any}{ \reflectbox {#2} }
742 {archaic}
743 {
744 \boustro_line_to_seq:n {#2}
745 \boustro_render_archaic_reverse:
746 }
747 {classical}
748 {
749 \boustro_line_to_seq:n {#2}
750 \boustro_render_classical_reverse:
751 }
752 }
753 { \reflectbox {#2} }
754 \group_end:
755 }
756
757 \cs_new_protected:Npn \boustro_typeset_inline:nn #1#2
758 {
759 \tl_set:Nn \__boustro_inline_mode_tl {#1}
760 \boustro_inline_parse:w #2 [ \q_no_value ] \q_stop
761 }
762
763 \cs_new_protected:Npn \boustro_inline_parse:w #1[#2]#3 \q_stop
764 {
765 #1
766 \quark_if_no_value:nF {#2}
767 {
768 \boustro_render_inline:nn { \__boustro_inline_mode_tl } {#2}
769 \boustro_inline_parse:w #3 \q_stop
770 }
771 }
772
773 % -----

```

```

774 % Public API (single command)
775 % -----
776 \NewDocumentCommand{\boustrophedon}{ O{} +m }
777 {
778   \group_begin:
779     \keys_set:nn { boustro } {#1}
780     \bool_if:NTF \g_boustro_inline_bool
781       { \exp_args:Nx \boustro_typeset_inline:nn { \tl_use:N \g_boustro_type_tl } {#2} }
782       { \exp_args:Nx \boustro_typeset_block:nn { \tl_use:N \g_boustro_type_tl } {#2} }
783     \group_end:
784   }
785
786 % Convenience aliases (v0.76 compatibility)
787 \NewDocumentCommand{\boustrophedonAny}{+m}
788 { \boustrophedon[Type=Any]{#1} }
789 \NewDocumentCommand{\boustrophedonArchaic}{+m}
790 { \boustrophedon[Type=Archaic]{#1} }
791 \NewDocumentCommand{\boustrophedonClassical}{+m}
792 { \boustrophedon[Type=Classical]{#1} }
793
794 % Block style API with start/stop commands (single command, optional [keys])
795 \makeatletter
796 \def\startBoustrophedon{%
797   \@ifnextchar[{\boustro_start_with_keys@}{\boustro_start_with_keys@[]}%
798 }
799 \long\def\boustro_start_with_keys@[#1]#2\stopBoustrophedon{%
800   \boustrophedon[#1]{#2}%
801 }
802 \makeatother
803
804 % -----
805 % Block style API with start/stop commands
806 % -----
807 \long\def\startBoustrophedonAny#1\stopBoustrophedonAny{\boustrophedonAny{#1}}
808 \long\def\startBoustrophedonArchaic#1\stopBoustrophedonArchaic{\boustrophedonArchaic{#1}}
809 \long\def\startBoustrophedonClassical#1\stopBoustrophedonClassical{\boustrophedonClassical{#1}}
810
811 % -----
812 % Engine notice
813 % -----
814 \AtBeginDocument{
815   \ifPDFTeX
816     \PackageWarningNoLine{boustrophedon}{Greek Unicode input works best with XeLaTeX or LuaLaTeX; transliteration input is still fine in pdfLaTeX}
817   \fi
818 }
819
820 \ExplSyntaxOff
821
822 </package>

```