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
```

```
\bool_new:N \g_boustro_obscure_letters_bool
\bool_new:N \g_boustro_reset_page_bool
\bool_new:N \g_boustro_inline_bool
\int_new:N  \g_boustro_line_int
\tl_new:N   \g_boustro_type_tl
\dim_new:N  \g_boustro_linewidth_dim

\bool_set_true:N  \g_boustro_start_ltr_bool
\bool_set_true:N  \g_boustro_obscure_letters_bool
\bool_set_true:N  \g_boustro_reset_page_bool
\bool_set_false:N \g_boustro_inline_bool
\int_gzero:N      \g_boustro_line_int
\tl_set:Nn        \g_boustro_type_tl {any}
\dim_set:Nn       \g_boustro_linewidth_dim { 0pt }

\cs_new_protected:Npn \boustro_set_linewidth:n #1
 {
  \tl_set:Nn \l_tmpb_tl {#1}
  \tl_trim_spaces:N \l_tmpb_tl
  \tl_set:NV \l_tmpa_tl \l_tmpb_tl
  \tl_set:Nx \l_tmpa_tl { \tl_lower_case:n { \l_tmpa_tl } }
  \tl_if_blank:VTF \l_tmpa_tl
    { \dim_set:Nn \g_boustro_linewidth_dim { 0pt } }
    {
      \str_if_eq:VnTF \l_tmpa_tl { auto }
        { \dim_set:Nn \g_boustro_linewidth_dim { 0pt } }
        {
          \dim_set:Nn \g_boustro_linewidth_dim { \l_tmpb_tl }
        }
    }
 }

\keys_define:nn { boustro }
 {
  start .choice:,
  start / LTR .code:n = { \bool_set_true:N  \g_boustro_start_ltr_bool },
  start / RTL .code:n = { \bool_set_false:N \g_boustro_start_ltr_bool },
  Type .choice:,
  Type / Any .code:n      = { \tl_set:Nn \g_boustro_type_tl {any} },
  Type / Archaic .code:n  = { \tl_set:Nn \g_boustro_type_tl {archaic} },
  Type / Classical .code:n= { \tl_set:Nn \g_boustro_type_tl {classical} },
  LineWidth .code:n = { \boustro_set_linewidth:n {#1} },
  ObscureLetters .choice:,
  ObscureLetters / On  .code:n = { \bool_set_true:N  \g_boustro_obscure_letters_bool },
  ObscureLetters / Off .code:n = { \bool_set_false:N \g_boustro_obscure_letters_bool },
  % Backwards-compat alias (deprecated):
  digammaAndKoppa .choice:,
  digammaAndKoppa / On  .code:n = { \bool_set_true:N  \g_boustro_obscure_letters_bool },
  digammaAndKoppa / Off .code:n = { \bool_set_false:N \g_boustro_obscure_letters_bool },
  resetDirection .choice:,
  resetDirection / Page      .code:n = { \bool_set_true:N  \g_boustro_reset_page_bool },
  resetDirection / Paragraph .code:n = { \bool_set_false:N \g_boustro_reset_page_bool },
  % Backwards-compat alias (deprecated):
  reset .choice:,
  reset / page      .code:n = { \bool_set_true:N  \g_boustro_reset_page_bool },
  reset / paragraph .code:n = { \bool_set_false:N \g_boustro_reset_page_bool },
  inLine .choice:,
  inLine / True  .code:n = { \bool_set_true:N  \g_boustro_inline_bool },
  inLine / False .code:n = { \bool_set_false:N \g_boustro_inline_bool },
  inLine / true  .code:n = { \bool_set_true:N  \g_boustro_inline_bool },
  inLine / false .code:n = { \bool_set_false:N \g_boustro_inline_bool },
  inline .choice:,
  inline / True  .code:n = { \bool_set_true:N  \g_boustro_inline_bool },
```

```
    inline / False .code:n = { \bool_set_false:N \g_boustro_inline_bool },
    inline / true  .code:n = { \bool_set_true:N  \g_boustro_inline_bool },
    inline / false .code:n = { \bool_set_false:N \g_boustro_inline_bool },
  }
\ProcessKeyOptions [ boustro ]

\AddToHook{shipout/after}
  {
    \bool_if:NT \g_boustro_reset_page_bool
      { \int_gzero:N \g_boustro_line_int }
  }

% ---------------------------------------------------------------------------
% Variables
% ---------------------------------------------------------------------------
\seq_new:N \l__boustro_lines_seq
\seq_new:N \l__boustro_symbols_seq
\seq_new:N \l__boustro_tmp_seq
\seq_new:N \l__boustro_words_seq
\seq_new:N \l__boustro_pars_seq
\bool_new:N \l__boustro_input_has_greek_bool
\tl_new:N  \l__boustro_norm_tl
\tl_new:N  \l__boustro_scan_tl
\tl_new:N  \l__boustro_scan_tail_tl
\tl_new:N  \l__boustro_head_tl
\tl_new:N  \l__boustro_next_tl
\tl_new:N  \l__boustro_combined_tl
\tl_new:N  \l__boustro_current_line_tl
\tl_new:N  \l__boustro_candidate_tl
\tl_new:N  \l__boustro_words_tl
\tl_new:N  \l__boustro_input_tl
\tl_new:N  \l__boustro_inline_mode_tl
\box_new:N \l__boustro_measure_box
\dim_new:N \l__boustro_width_dim
\dim_new:N \l__boustro_target_dim

\cs_new_protected:Npn \boustro_set_target_width:n #1
  {
    \dim_compare:nNnTF { \g_boustro_linewidth_dim } > { 0pt }
      { \dim_set_eq:NN \l__boustro_target_dim \g_boustro_linewidth_dim }
      {
        \str_case:nnF {#1}
          {
            {any}{\dim_set:Nn \l__boustro_target_dim { .95\linewidth }}
            {archaic}{\dim_set:Nn \l__boustro_target_dim { .95\linewidth }}
            {classical}{\dim_set:Nn \l__boustro_target_dim { .95\linewidth }}
          }
          { \dim_set:Nn \l__boustro_target_dim { .95\linewidth } }
      }
    \dim_compare:nNnT { \l__boustro_target_dim } > { \linewidth }
      { \dim_set:Nn \l__boustro_target_dim { \linewidth } }
  }

\cs_new_protected:Npn \boustro_maybe_reset:
  {
    \bool_if:NF \g_boustro_reset_page_bool
      { \int_gzero:N \g_boustro_line_int }
  }

\cs_new:Npn \boustro_line_is_reverse:
  {
    \bool_if:NTF \g_boustro_start_ltr_bool
      { \int_if_even_p:n { \g_boustro_line_int } }
```

```
    { \int_if_odd_p:n  { \g_boustro_line_int } }
  }

\cs_new:Npn \boustro_line_is_reverse_n:n #1
  {
    \bool_if:NTF \g_boustro_start_ltr_bool
      { \int_if_even_p:n {#1} }
      { \int_if_odd_p:n  {#1} }
  }

% ---------------------------------------------------------------------------
% Explicit line splitting and paragraph splitting
% ---------------------------------------------------------------------------
\cs_new_protected:Npn \boustro_split_explicit_lines:n #1
  {
    \tl_set:Nn \l__boustro_input_tl {#1}
    \tl_replace_all:Nnn \l__boustro_input_tl {\\} { <BLINEBREAK> }
    \seq_clear:N \l__boustro_pars_seq
    \seq_set_split:NnV \l__boustro_tmp_seq { <BLINEBREAK> } \l__boustro_input_tl
    \seq_map_inline:Nn \l__boustro_tmp_seq
      {
        \tl_set:Nn \l_tmpa_tl {##1}
        \tl_trim_spaces:N \l_tmpa_tl
        \tl_if_blank:VF \l_tmpa_tl
          { \seq_put_right:NV \l__boustro_pars_seq \l_tmpa_tl }
      }
  }

% ---------------------------------------------------------------------------
% Unicode Greek normalization to internal symbols.
% Sentinels: 1 theta, 2 phi, 3 chi, 4 psi, 5 xi, 6 omega, 7 koppa, 8 digamma
% ---------------------------------------------------------------------------
\cs_new_protected:Npn \boustro_normalize_unicode_greek:
  {
    % remove common accent marks first
    \regex_replace_all:nnN { [άὰᾶἀἁᾄᾴᾅᾲᾳᾈᾀᾁᾂᾃᾆᾇᾰᾱᾌᾋᾍᾎᾏᾺᾼᾹᾸ] } { a } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ἄἂἃἅἆἇᾬᾫᾭᾯᾪᾩᾧᾦᾥᾤᾣᾢᾡᾠᾷᾶ᾵ᾄᾂᾁᾀᾃᾅᾆᾇᾈᾉᾊᾋᾌᾍᾎᾏᾲᾴᾳᾼᾺᾹᾸᾰᾱᾀᾁᾂᾃᾄᾅᾆᾇᾈᾉᾊᾋᾌᾍᾎᾏ] } { A } \l__boustro_norm_tl
    \regex_replace_all:nnN { [έὲἐἑἔἕ] } { e } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ἝἜἙἚἛἘΈ] } { E } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ήὴῆἠἡᾖᾔᾐᾑᾗᾕᾓῄῂῇᾘᾙᾚᾛᾜᾝᾞᾟ] } { h } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ΉῊἨἩἪἫἬἭᾘᾙᾚᾛᾜᾝᾞᾟ] } { H } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ίὶῖϊΐἰἱἴἵἶἷῒΐῗ] } { i } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ΊΙΪἸἹἼἽἾἿΙ] } { I } \l__boustro_norm_tl
    \regex_replace_all:nnN { [όὸὀὁὄὅ] } { o } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ΌΟΌὈὉὊὋὌ] } { O } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ύὺῦϋΰὐὑὔὕὖὗ] } { u } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ΎΥΫὙὝὛὟ] } { U } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ώὼῶωὠὡὤὥὦὧᾠᾡᾤᾣᾦᾧῲῳῴῶῷῺΏ] } { 6 } \l__boustro_norm_tl
    \regex_replace_all:nnN { [ΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩΩ] } { 6 } \l__boustro_norm_tl

    \bool_if:NT \g_boustro_obscure_letters_bool
      {
        % Unicode koppa / digamma (if user inputs them directly)
        \regex_replace_all:nnN { Ϙ|ϙ } { 7 } \l__boustro_norm_tl
        \regex_replace_all:nnN { F|ϝ } { 8 } \l__boustro_norm_tl
      }

    % plain Greek letters
    \regex_replace_all:nnN { α } { a } \l__boustro_norm_tl
    \regex_replace_all:nnN { A } { A } \l__boustro_norm_tl
    \regex_replace_all:nnN { β } { b } \l__boustro_norm_tl
    \regex_replace_all:nnN { B } { B } \l__boustro_norm_tl
    \regex_replace_all:nnN { γ } { g } \l__boustro_norm_tl
```

```
  \regex_replace_all:nnN { Γ } { G } \l__boustro_norm_tl
  \regex_replace_all:nnN { δ } { d } \l__boustro_norm_tl
  \regex_replace_all:nnN { Δ } { D } \l__boustro_norm_tl
  \regex_replace_all:nnN { ε } { e } \l__boustro_norm_tl
  \regex_replace_all:nnN { E } { E } \l__boustro_norm_tl
  \regex_replace_all:nnN { ζ } { z } \l__boustro_norm_tl
  \regex_replace_all:nnN { Z } { Z } \l__boustro_norm_tl
  \regex_replace_all:nnN { η } { h } \l__boustro_norm_tl
  \regex_replace_all:nnN { H } { H } \l__boustro_norm_tl
  \regex_replace_all:nnN { θ } { 1 } \l__boustro_norm_tl
  \regex_replace_all:nnN { Θ } { 1 } \l__boustro_norm_tl
  \regex_replace_all:nnN { ι|ϊ|ΐ } { i } \l__boustro_norm_tl
  \regex_replace_all:nnN { I|Ϊ } { I } \l__boustro_norm_tl
  \regex_replace_all:nnN { κ } { k } \l__boustro_norm_tl
  \regex_replace_all:nnN { K } { K } \l__boustro_norm_tl
  \regex_replace_all:nnN { λ } { l } \l__boustro_norm_tl
  \regex_replace_all:nnN { Λ } { L } \l__boustro_norm_tl
  \regex_replace_all:nnN { μ } { m } \l__boustro_norm_tl
  \regex_replace_all:nnN { M } { M } \l__boustro_norm_tl
  \regex_replace_all:nnN { ν } { n } \l__boustro_norm_tl
  \regex_replace_all:nnN { N } { N } \l__boustro_norm_tl
  \regex_replace_all:nnN { ξ } { 5 } \l__boustro_norm_tl
  \regex_replace_all:nnN { Ξ } { 5 } \l__boustro_norm_tl
  \regex_replace_all:nnN { ο } { o } \l__boustro_norm_tl
  \regex_replace_all:nnN { O } { O } \l__boustro_norm_tl
  \regex_replace_all:nnN { π } { p } \l__boustro_norm_tl
  \regex_replace_all:nnN { Π } { P } \l__boustro_norm_tl
  \regex_replace_all:nnN { ρ } { r } \l__boustro_norm_tl
  \regex_replace_all:nnN { P } { R } \l__boustro_norm_tl
  \regex_replace_all:nnN { σ|ς } { s } \l__boustro_norm_tl
  \regex_replace_all:nnN { Σ } { S } \l__boustro_norm_tl
  \regex_replace_all:nnN { τ } { t } \l__boustro_norm_tl
  \regex_replace_all:nnN { T } { T } \l__boustro_norm_tl
  \regex_replace_all:nnN { υ|ϋ|ΰ } { u } \l__boustro_norm_tl
  \regex_replace_all:nnN { Y|Ÿ } { U } \l__boustro_norm_tl
  \regex_replace_all:nnN { φ } { 2 } \l__boustro_norm_tl
  \regex_replace_all:nnN { Φ } { 2 } \l__boustro_norm_tl
  \regex_replace_all:nnN { χ } { 3 } \l__boustro_norm_tl
  \regex_replace_all:nnN { X } { 3 } \l__boustro_norm_tl
  \regex_replace_all:nnN { ψ } { 4 } \l__boustro_norm_tl
  \regex_replace_all:nnN { Ψ } { 4 } \l__boustro_norm_tl
  \regex_replace_all:nnN { ω } { 6 } \l__boustro_norm_tl
  \regex_replace_all:nnN { Ω } { 6 } \l__boustro_norm_tl
}

\cs_new_protected:Npn \boustro_normalize_input:n #1
{
  \tl_set:Nn \l__boustro_norm_tl {#1}
  \regex_match:nnTF { [\x{0370}-\x{03FF}\x{1F00}-\x{1FFF}] } {#1}
    { \bool_set_true:N \l__boustro_input_has_greek_bool }
    { \bool_set_false:N \l__boustro_input_has_greek_bool }
  \boustro_normalize_unicode_greek:

  \bool_if:NF \l__boustro_input_has_greek_bool
    {
      \regex_replace_all:nnN { J } { I } \l__boustro_norm_tl
      \regex_replace_all:nnN { j } { i } \l__boustro_norm_tl
      \regex_replace_all:nnN { Ō|ō|Ô|ô } { 6 } \l__boustro_norm_tl
      \bool_if:NT \g_boustro_obscure_letters_bool
        {
          \regex_replace_all:nnN { TH|Th|tH|th } { 1 } \l__boustro_norm_tl
          \regex_replace_all:nnN { PH|Ph|pH|ph } { 2 } \l__boustro_norm_tl
          \regex_replace_all:nnN { CH|Ch|cH|ch|KH|Kh|kH|kh } { 3 } \l__boustro_norm_tl
```

```
        \regex_replace_all:nnN { PS|Ps|pS|ps } { 4 } \l__boustro_norm_tl
        \regex_replace_all:nnN { KS|Ks|kS|ks } { 5 } \l__boustro_norm_tl
        \regex_replace_all:nnN { KA } { 7A } \l__boustro_norm_tl
        \regex_replace_all:nnN { Ka } { 7a } \l__boustro_norm_tl
        \regex_replace_all:nnN { kA } { 7A } \l__boustro_norm_tl
        \regex_replace_all:nnN { ka } { 7a } \l__boustro_norm_tl
        \regex_replace_all:nnN { KO } { 7O } \l__boustro_norm_tl
        \regex_replace_all:nnN { Ko } { 7o } \l__boustro_norm_tl
        \regex_replace_all:nnN { kO } { 7O } \l__boustro_norm_tl
        \regex_replace_all:nnN { ko } { 7o } \l__boustro_norm_tl
        \regex_replace_all:nnN { W } { 8 } \l__boustro_norm_tl
        \regex_replace_all:nnN { w } { 8 } \l__boustro_norm_tl
      }
    }

    \regex_replace_all:nnN { [.,;:!?] } { ~ } \l__boustro_norm_tl
    \regex_replace_all:nnN { [-()\[\]"'] } { ~ } \l__boustro_norm_tl
    \regex_replace_all:nnN { [^A-Za-z0-9~ ] } { } \l__boustro_norm_tl
    % Normalize any remaining whitespace to a visible/measureable separator.
    % Using `~` lets us map it to an explicit `\space` during rendering so it
    % isn't lost inside macro expansions (important for width-based wrapping).
    \regex_replace_all:nnN { \s+ } { ~ } \l__boustro_norm_tl
    \regex_replace_all:nnN { ~+ } { ~ } \l__boustro_norm_tl
  }

\cs_new_protected:Npn \boustro_line_to_seq:n #1
  {
    \seq_clear:N \l__boustro_symbols_seq
    \boustro_normalize_input:n {#1}
    % Treat certain digraph encodings as atomic tokens (e.g. 7A/7O for koppa+vowel)
    % so reversal doesn't split them into separate characters.
    \tl_set_eq:NN \l__boustro_scan_tl \l__boustro_norm_tl
    \boustro_scan_norm_into_symbols:
  }

\cs_new_protected:Npn \boustro_scan_norm_into_symbols:
  {
    \tl_if_empty:NTF \l__boustro_scan_tl
      { }
      {
        \tl_set:Nx \l__boustro_head_tl { \tl_head:N \l__boustro_scan_tl }
        \tl_set:Nx \l__boustro_scan_tail_tl { \tl_tail:N \l__boustro_scan_tl }

        \tl_if_empty:NTF \l__boustro_scan_tail_tl
          {
            \seq_put_right:NV \l__boustro_symbols_seq \l__boustro_head_tl
            \tl_clear:N \l__boustro_scan_tl
          }
          {
            \tl_set:Nx \l__boustro_next_tl { \tl_head:N \l__boustro_scan_tail_tl }
            \tl_if_eq:NnTF \l__boustro_head_tl { 7 }
              {
                \str_case:nnF { \l__boustro_next_tl }
                  {
                    {A}{ \tl_set:Nn \l__boustro_combined_tl { 7A } }
                    {a}{ \tl_set:Nn \l__boustro_combined_tl { 7a } }
                    {O}{ \tl_set:Nn \l__boustro_combined_tl { 7O } }
                    {o}{ \tl_set:Nn \l__boustro_combined_tl { 7o } }
                  }
                  { \tl_clear:N \l__boustro_combined_tl }

                \tl_if_blank:VTF \l__boustro_combined_tl
                  {
```

```
                \seq_put_right:NV \l__boustro_symbols_seq \l__boustro_head_tl
                \tl_set_eq:NN \l__boustro_scan_tl \l__boustro_scan_tail_tl
              }
              {
                \seq_put_right:NV \l__boustro_symbols_seq \l__boustro_combined_tl
                \tl_set:Nx \l__boustro_scan_tl { \tl_tail:N \l__boustro_scan_tail_tl }
              }
            }
            {
              \seq_put_right:NV \l__boustro_symbols_seq \l__boustro_head_tl
              \tl_set_eq:NN \l__boustro_scan_tl \l__boustro_scan_tail_tl
            }
        }
      \boustro_scan_norm_into_symbols:
    }
}

% ---------------------------------------------------------------------
% Glyph mapping helpers
% ---------------------------------------------------------------------
\cs_new:Npn \boustro_archaic_map:n #1
 {
  \str_case:nnF {#1}
   {
    {~}{\space}
    {A}{\Aalpha}{a}{\Aalpha}
    {B}{\Abeta}{b}{\Abeta}
    {G}{\Agamma}{g}{\Agamma}
    {D}{\Adelta}{d}{\Adelta}
    {E}{\Aepsilon}{e}{\Aepsilon}
    {Z}{\Azeta}{z}{\Azeta}
    {H}{\Aeta}{h}{\Aeta}
    {1}{\Atheta}
    {I}{\Aiota}{i}{\Aiota}
    {K}{\Akappa}{k}{\Akappa}
    {7}{\Akoppa}
    {7A}{\Akoppa\Aalpha}{7a}{\Akoppa\Aalpha}
    {7O}{\Akoppa\Aomicron}{7o}{\Akoppa\Aomicron}
    {L}{\Alambda}{l}{\Alambda}
    {M}{\Amu}{m}{\Amu}
    {N}{\Anu}{n}{\Anu}
    {X}{\Axi}{x}{\Axi}{5}{\Axi}
    {O}{\Aomicron}{o}{\Aomicron}
    {6}{\Aomega}
    {P}{\Api}{p}{\Api}
    {R}{\Arho}{r}{\Arho}
    {S}{\Asigma}{s}{\Asigma}
    {T}{\Atau}{t}{\Atau}
    {U}{\Aupsilon}{u}{\Aupsilon}
    {Y}{\Aupsilon}{y}{\Aupsilon}
    {3}{\Achi}
    {2}{\Aphi}
    {4}{\Apsi}
    {8}{\Adigamma}
   }
   {#1}
 }

\cs_new:Npn \boustro_classical_normal_map:n #1
 {
  \str_case:nnF {#1}
   {
    {~}{\space}
```

```
      {A}{\Aalpha}{a}{\Aalpha}
      {B}{\Abeta}{b}{\Abeta}
      {G}{\Agamma}{g}{\Agamma}
      {D}{\Adelta}{d}{\Adelta}
      {E}{\Aepsilon}{e}{\Aepsilon}
      {Z}{\Azeta}{z}{\Azeta}
      {H}{\Aeta}{h}{\Aeta}
      {1}{\Atheta}
      {I}{\Aiota}{i}{\Aiota}
      {K}{\Akappa}{k}{\Akappa}
      {L}{\Alambda}{l}{\Alambda}
      {M}{\Amu}{m}{\Amu}
      {N}{\Anu}{n}{\Anu}
      {X}{\Axi}{x}{\Axi}{5}{\Axi}
      {O}{\Aomicron}{o}{\Aomicron}
      {6}{\Aomega}
      {P}{\Api}{p}{\Api}
      {R}{\Arho}{r}{\Arho}
      {S}{\Asigma}{s}{\Asigma}
      {T}{\Atau}{t}{\Atau}
      {U}{\Aupsilon}{u}{\Aupsilon}
      {Y}{\Aupsilon}{y}{\Aupsilon}
      {3}{\Achi}
      {2}{\Aphi}
      {4}{\Apsi}
    }
    {
      \str_if_eq:nnTF {#1}{7A}
        { \textgvibc{\Akoppa}\Aalpha }
        {
          \str_if_eq:nnTF {#1}{7a}
            { \textgvibc{\Akoppa}\Aalpha }
            {
              \str_if_eq:nnTF {#1}{7O}
                { \textgvibc{\Akoppa}\Aomicron }
                {
                  \str_if_eq:nnTF {#1}{7o}
                    { \textgvibc{\Akoppa}\Aomicron }
                    {
      \str_if_eq:nnTF {#1}{7}
        {\textgvibc{\Akoppa}}
        {
          \str_if_eq:nnTF {#1}{8}
            {\textgvibc{\Adigamma}}
            {#1}
        }
                    }
                }
            }
        }
    }
  }

\cs_new:Npn \boustro_classical_reverse_map:n #1
  {
    \str_case:nnF {#1}
      {
        {~}{\space}
        {A}{\ARalpha}{a}{\ARalpha}
        {B}{\ARbeta}{b}{\ARbeta}
        {G}{\ARgamma}{g}{\ARgamma}
        {D}{\ARdelta}{d}{\ARdelta}
        {E}{\ARepsilon}{e}{\ARepsilon}
```

```
      {Z}{\ARzeta}{z}{\ARzeta}
      {H}{\AReta}{h}{\AReta}
      {1}{\ARtheta}
      {I}{\ARiota}{i}{\ARiota}
      {K}{\ARkappa}{k}{\ARkappa}
      {L}{\ARlambda}{l}{\ARlambda}
      {M}{\ARmu}{m}{\ARmu}
      {N}{\ARnu}{n}{\ARnu}
      {X}{\ARxi}{x}{\ARxi}{5}{\ARxi}
      {O}{\ARomicron}{o}{\ARomicron}
      {6}{\ARomega}
      {P}{\ARpi}{p}{\ARpi}
      {R}{\ARrho}{r}{\ARrho}
      {S}{\ARsigma}{s}{\ARsigma}
      {T}{\ARtau}{t}{\ARtau}
      {U}{\ARupsilon}{u}{\ARupsilon}
      {Y}{\ARupsilon}{y}{\ARupsilon}
      {3}{\ARchi}
      {2}{\ARphi}
      {4}{\ARpsi}
    }
    {
      \str_if_eq:nnTF {#1}{7A}
        { \textgvibc{\Akoppa}\ARalpha }
        {
          \str_if_eq:nnTF {#1}{7a}
            { \textgvibc{\Akoppa}\ARalpha }
            {
              \str_if_eq:nnTF {#1}{7O}
                { \textgvibc{\Akoppa}\ARomicron }
                {
                  \str_if_eq:nnTF {#1}{7o}
                    { \textgvibc{\Akoppa}\ARomicron }
                    {
      \str_if_eq:nnTF {#1}{7}
        {\textgvibc{\Akoppa}}
        {
          \str_if_eq:nnTF {#1}{8}
            {\textgvibc{\Adigamma}}
            {#1}
        }
                    }
                }
            }
        }
    }
  }

\cs_new_protected:Npn \boustro_render_archaic_forward:
  { {\textgvibc{\seq_map_inline:Nn \l__boustro_symbols_seq {\boustro_archaic_map:n{##1}}}} }
\cs_new_protected:Npn \boustro_render_archaic_reverse:
  {
    \seq_set_eq:NN \l__boustro_tmp_seq \l__boustro_symbols_seq
    \seq_reverse:N \l__boustro_tmp_seq
    {
      \textgvibc
        {
          \seq_map_inline:Nn \l__boustro_tmp_seq
            {
              \str_if_eq:nnTF {##1} {~}
                { \space }
                { \reflectbox { \boustro_archaic_map:n {##1} } }
            }
```

```
      }
    }
  }
\cs_new_protected:Npn \boustro_render_classical_forward:
  { {\textgivbc{\seq_map_inline:Nn \l__boustro_symbols_seq {\boustro_classical_normal_map:n{##1}}}} }
\cs_new_protected:Npn \boustro_render_classical_reverse:
  {
    \seq_set_eq:NN \l__boustro_tmp_seq \l__boustro_symbols_seq
    \seq_reverse:N \l__boustro_tmp_seq
    {
      \textgivbc
      {
        \seq_map_inline:Nn \l__boustro_tmp_seq
          {
            \str_if_eq:nnTF {##1} {~}
              { \space }
              { \reflectbox { \boustro_classical_normal_map:n {##1} } }
          }
      }
    }
  }

% ---------------------------------------------------------------------------
% Width measurement and wrapping
% ---------------------------------------------------------------------------
\cs_new_protected:Npn \boustro_measure_any:nN #1#2
  {
    \hbox_set:Nn \l__boustro_measure_box {#1}
    \dim_set:Nn #2 { \box_wd:N \l__boustro_measure_box }
  }

\cs_new_protected:Npn \boustro_measure_archaic:nN #1#2
  {
    \boustro_line_to_seq:n {#1}
    \hbox_set:Nn \l__boustro_measure_box { \boustro_render_archaic_forward: }
    \dim_set:Nn #2 { \box_wd:N \l__boustro_measure_box }
  }

\cs_new_protected:Npn \boustro_measure_classical:nN #1#2
  {
    \boustro_line_to_seq:n {#1}
    \hbox_set:Nn \l__boustro_measure_box { \boustro_render_classical_forward: }
    \dim_set:Nn #2 { \box_wd:N \l__boustro_measure_box }
  }
\cs_generate_variant:Nn \boustro_measure_any:nN { VN }
\cs_generate_variant:Nn \boustro_measure_archaic:nN { VN }
\cs_generate_variant:Nn \boustro_measure_classical:nN { VN }

\cs_new:Npn \boustro_map_string_archaic:n #1
  {
    \boustro_line_to_seq:n {#1}
    \seq_map_inline:Nn \l__boustro_symbols_seq { \boustro_archaic_map:n{##1} }
  }

\cs_new:Npn \boustro_map_string_classical:n #1
  {
    \boustro_line_to_seq:n {#1}
    \seq_map_inline:Nn \l__boustro_symbols_seq { \boustro_classical_normal_map:n{##1} }
  }

\cs_new_protected:Npn \boustro_wrap_text:nnN #1#2#3
  {
    \seq_clear:N #3
```

```
  \boustro_set_target_width:n {#1}
  \tl_set:Nx \l__boustro_words_tl { \tl_to_str:n {#2} }
  \regex_replace_all:nnN { \s+ } { <BSPACE> } \l__boustro_words_tl
  \seq_set_split:NnV \l__boustro_words_seq { <BSPACE> } \l__boustro_words_tl
  \seq_remove_all:Nn \l__boustro_words_seq { }
  \tl_clear:N \l__boustro_current_line_tl
  \seq_map_inline:Nn \l__boustro_words_seq
    {
      \tl_if_blank:VTF \l__boustro_current_line_tl
        { \tl_set:Nn \l__boustro_candidate_tl {##1} }
        { \tl_set:Ne \l__boustro_candidate_tl { \l__boustro_current_line_tl \c_space_tl ##1 } }
      \use:c { boustro_measure_#1:VN } \l__boustro_candidate_tl \l__boustro_width_dim
      \dim_compare:nNnTF { \l__boustro_width_dim } > { \l__boustro_target_dim }
        {
          \tl_if_blank:VTF \l__boustro_current_line_tl
            { \seq_put_right:Nn #3 {##1} }
            {
              \seq_put_right:NV #3 \l__boustro_current_line_tl
              \tl_set:Nn \l__boustro_current_line_tl {##1}
            }
        }
        { \tl_set:NV \l__boustro_current_line_tl \l__boustro_candidate_tl }
    }
  \tl_if_blank:VF \l__boustro_current_line_tl
    { \seq_put_right:NV #3 \l__boustro_current_line_tl }
}

\cs_new_protected:Npn \boustro_wrap_epigraphic_text:nnN #1#2#3
{
  \seq_clear:N #3
  \boustro_normalize_input:n {#2}
  \tl_set_eq:NN \l__boustro_words_tl \l__boustro_norm_tl
  \regex_replace_all:nnN { \s+ } { <BSPACE> } \l__boustro_words_tl
  \seq_set_split:NnV \l__boustro_words_seq { <BSPACE> } \l__boustro_words_tl
  \seq_remove_all:Nn \l__boustro_words_seq { }
  \boustro_set_target_width:n {#1}
  \tl_clear:N \l__boustro_current_line_tl
  \seq_map_inline:Nn \l__boustro_words_seq
    {
      \tl_if_blank:VTF \l__boustro_current_line_tl
        { \tl_set:Nn \l__boustro_candidate_tl {##1} }
        { \tl_set:Ne \l__boustro_candidate_tl { \l__boustro_current_line_tl \c_space_tl ##1 } }
      \use:c { boustro_measure_#1:VN } \l__boustro_candidate_tl \l__boustro_width_dim
      \dim_compare:nNnTF { \l__boustro_width_dim } > { \l__boustro_target_dim }
        {
          \tl_if_blank:VTF \l__boustro_current_line_tl
            { \seq_put_right:Nn #3 {##1} }
            {
              \seq_put_right:NV #3 \l__boustro_current_line_tl
              \tl_set:Nn \l__boustro_current_line_tl {##1}
            }
        }
        { \tl_set:NV \l__boustro_current_line_tl \l__boustro_candidate_tl }
    }
  \tl_if_blank:VF \l__boustro_current_line_tl
    { \seq_put_right:NV #3 \l__boustro_current_line_tl }
}

\cs_new_protected:Npn \boustro_prepare_lines:nn #1#2
{
  \seq_clear:N \l__boustro_lines_seq
  \tl_set:Nn \l__boustro_input_tl {#2}
  \tl_if_in:NnTF \l__boustro_input_tl {\\}
```

```
    {
      \boustro_split_explicit_lines:n {#2}
      \seq_map_inline:Nn \l__boustro_pars_seq
        {
          \boustro_wrap_text:nnN {#1} {##1} \l__boustro_tmp_seq
          \seq_map_inline:Nn \l__boustro_tmp_seq
            { \seq_put_right:Nn \l__boustro_lines_seq {####1} }
        }
    }
    {
      \str_case:nnF {#1}
        {
          {any}{\boustro_wrap_text:nnN {any} {#2} \l__boustro_lines_seq}
          {archaic}{\boustro_wrap_text:nnN {archaic} {#2} \l__boustro_lines_seq}
          {classical}{\boustro_wrap_text:nnN {classical} {#2} \l__boustro_lines_seq}
        }
        { \boustro_wrap_text:nnN {#1} {#2} \l__boustro_lines_seq }
    }
  }

% ---------------------------------------------------------------------------
% Printing line by line
% ---------------------------------------------------------------------------
\cs_new_protected:Npn \boustro_print_any_line:nn #1#2
  {
    \par\noindent
    \bool_if:nTF { \boustro_line_is_reverse_n:n {#1} }
      { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][r]{\reflectbox{#2}}}\par }
      { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][l]{#2}}\par }
  }

\cs_new_protected:Npn \boustro_print_archaic_line:nn #1#2
  {
    \par\noindent
    \boustro_line_to_seq:n {#2}
    \bool_if:nTF { \boustro_line_is_reverse_n:n {#1} }
      { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][r]{\boustro_render_archaic_reverse:}}\par }
      { \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][l]{\boustro_render_archaic_forward:}}\par }
  }

\cs_new_protected:Npn \boustro_print_classical_line:nn #1#2
  {
    \par\noindent
    \boustro_line_to_seq:n {#2}
    \bool_if:nTF { \boustro_line_is_reverse_n:n {#1} }
      {
        \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][r]{\boustro_render_classical_reverse:}}\par
      }
      {
        \makebox[\linewidth][c]{\makebox[\l__boustro_target_dim][l]{\boustro_render_classical_forward:}}\par
      }
  }

% ---------------------------------------------------------------------------
% Public typesetter (mode, text)
% ---------------------------------------------------------------------------
\cs_new_protected:Npn \boustro_typeset_block:nn #1#2
  {
    \boustro_maybe_reset:
    \boustro_set_target_width:n {#1}
    \boustro_prepare_lines:nn {#1} {#2}
    \int_set:Nn \l_tmpa_int { \g_boustro_line_int }
    \str_case:nnF {#1}
```

```
    {
      {any}
        {
          \seq_map_indexed_inline:Nn \l__boustro_lines_seq
            { \boustro_print_any_line:nn { \int_eval:n { \l_tmpa_int + ##1 } } {##2} }
        }
      {archaic}
        {
          \seq_map_indexed_inline:Nn \l__boustro_lines_seq
            { \boustro_print_archaic_line:nn { \int_eval:n { \l_tmpa_int + ##1 } } {##2} }
        }
      {classical}
        {
          \seq_map_indexed_inline:Nn \l__boustro_lines_seq
            { \boustro_print_classical_line:nn { \int_eval:n { \l_tmpa_int + ##1 } } {##2} }
        }
    }{}
  \int_gadd:Nn \g_boustro_line_int { \seq_count:N \l__boustro_lines_seq }
  \par
}

% ---------------------------------------------------------------------------
% Inline markup: normal text outside [ ... ], boustrophedon only inside.
% The square brackets are markup delimiters and are not printed.
% ---------------------------------------------------------------------------
\cs_new_protected:Npn \boustro_render_inline:nn #1#2
{
  \group_begin:
    \str_case:VnF \l__boustro_inline_mode_tl
      {
        {any}{ \reflectbox {#2} }
        {archaic}
          {
            \boustro_line_to_seq:n {#2}
            \boustro_render_archaic_reverse:
          }
        {classical}
          {
            \boustro_line_to_seq:n {#2}
            \boustro_render_classical_reverse:
          }
      }
      { \reflectbox {#2} }
  \group_end:
}

\cs_new_protected:Npn \boustro_typeset_inline:nn #1#2
{
  \tl_set:Nn \l__boustro_inline_mode_tl {#1}
  \boustro_inline_parse:w #2 [ \q_no_value ] \q_stop
}

\cs_new_protected:Npn \boustro_inline_parse:w #1[#2]#3 \q_stop
{
  #1
  \quark_if_no_value:nF {#2}
    {
      \boustro_render_inline:nn { \l__boustro_inline_mode_tl } {#2}
      \boustro_inline_parse:w #3 \q_stop
    }
}

% ---------------------------------------------------------------------------
```

```
% Public API (single command)
% ---------------------------------------------------------------------------
\NewDocumentCommand{\boustrophedon}{ O{} +m }
  {
    \group_begin:
      \keys_set:nn { boustro } {#1}
      \bool_if:NTF \g_boustro_inline_bool
        { \exp_args:Nx \boustro_typeset_inline:nn { \tl_use:N \g_boustro_type_tl } {#2} }
        { \exp_args:Nx \boustro_typeset_block:nn { \tl_use:N \g_boustro_type_tl } {#2} }
    \group_end:
  }

% Convenience aliases (v0.76 compatibility)
\NewDocumentCommand{\boustrophedonAny}{+m}
  { \boustrophedon[Type=Any]{#1} }
\NewDocumentCommand{\boustrophedonArchaic}{+m}
  { \boustrophedon[Type=Archaic]{#1} }
\NewDocumentCommand{\boustrophedonClassical}{+m}
  { \boustrophedon[Type=Classical]{#1} }

% Block style API with start/stop commands (single command, optional [keys])
\makeatletter
\def\startBoustrophedon{%
  \@ifnextchar[{\boustro_start_with_keys@}{\boustro_start_with_keys@[]}%
}
\long\def\boustro_start_with_keys@[#1]#2\stopBoustrophedon{%
  \boustrophedon[#1]{#2}%
}
\makeatother

% ---------------------------------------------------------------------------
% Block style API with start/stop commands
% ---------------------------------------------------------------------------
\long\def\startBoustrophedonAny#1\stopBoustrophedonAny{\boustrophedonAny{#1}}
\long\def\startBoustrophedonArchaic#1\stopBoustrophedonArchaic{\boustrophedonArchaic{#1}}
\long\def\startBoustrophedonClassical#1\stopBoustrophedonClassical{\boustrophedonClassical{#1}}

% ---------------------------------------------------------------------------
% Engine notice
% ---------------------------------------------------------------------------
\AtBeginDocument{
  \ifPDFTeX
    \PackageWarningNoLine{boustrophedon}{Greek Unicode input works best with XeLaTeX or LuaLaTeX; transliteration input is still fine in pdfLaTeX}
  \fi
}

\ExplSyntaxOff

⟨/package⟩
```