# Unicode Alphabets for LaTeX

Mikkel Eide Eriksen
mikkel.eriksen@gmail.com

July 22, 2019

## 1 Preface

While Unicode supports the vast majority of use cases, there are certain specialized niches which require characters and glyphs not (yet) represented in the standard.

Thus the Private Use Area (PUA) at code points `E000–F8FF`, which enables third parties to define arbitrary character sets.

This package allows configuring a number of macros to enter characters from the PUA by name or code point.

## 2 Setup

The package is configured in the following manner:

```
\usepackage[options]{unicode-alphabets}
```

Where `options` must be one or more of the following character sets. See references for further detail on each, as well as usable fonts.

**AGL** Adobe Glyph List[1].

**CYFI** Early Cyrillic glyphs[2].

**LINCUA** Shortcut for enabling the character sets CYFI, MUFI, and TITUS[6].

**MUFI** The Medieval Unicode Font Initiative[4].

**SIL** SIL International[3].

**TITUS** Thesaurus Indogermanischer Text- und Sprachmaterialien[7].

**UCSUR** Under-ConScript Unicode Registry[8].

**UNZ** Normung von Sonderzeichen[5].

There is no default, since future versions of this package may supply more character sets that are incompatible with the above.

Additionally, one may configure different fonts for each character set, as in the following example:

```
\usepackage[
        MUFI,
        TITUS,
        fonts={
                MUFI=\mufifont,
                TITUS=\titusfont,
        },
]{unicode-alphabets}
```

Figure 1: Example setup

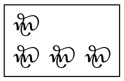
If no fonts are configured, the document font will be used (note that this may give undesirable results, as few fonts support multiple character sets[1]).

Finally, the `MUFI` and `UNZ` character sets have defined entity names, which result in the creation of macros for each character (see `msignfour` in the following example). These can be suppressed with the `disableentitymacros` option.

# 3 Usage

Each set defines a macro in the following manner. Let's use `MUFI` as an example.

By default, a macro with the lower-case name of the character set is defined: `\mufi{}`. It can then be used to display characters from the given set (the below uses the *Palemonas MUFI* font available from the MUFI project):



```
\mufi{FLOURISHED SMALL LETTER M SIGN}\\
\msignflour{}
\mufi{msignflour}
\mufi{F2F3}
```

Figure 2: Example usage

Additionally, starred versions of each macro are defined, which suppress using the configured font, falling back to the document font.

[1] I believe Andreas Stötzner's *Andron Mega* does, albeit I haven't tried it as it is somewhat expensive.

# References

[1] *adobe-type-tools/agl-aglfn*. URL: `https://github.com/adobe-type-tools/agl-aglfn/blob/master/glyphlist.txt` (visited on July 20, 2019).

[2] Victor Baranov et al. “Proposal for a unified encoding of Early Cyrillic glyphs in the Unicode Private Use Area”. In: *Scripta & e-scripta* 8 (2010), pp. 9–26.

[3] Lorna Evans. *SIL PUA 9.0*. July 5, 2016. URL: `https://scripts.sil.org/SILPUAassignments`.

[4] *MUFI: The Medieval Unicode Font Initiative*. URL: `https://mufi.info/` (visited on July 12, 2019).

[5] Bund für deutsche Schrift und Sprache e. V. *Normung von Sonderzeichen*. URL: `https://www.bfds.de/der-bund/normung-von-sonderzeichen-unicode/` (visited on July 12, 2019).

[6] Andreas Stötzner. “LINCUA: A Unicode-PUA harmonization scheme”. In: (2012).

[7] *Thesaurus Indogermanischer Text- und Sprachmaterialien*. July 10, 2017. URL: `http://titus.uni-frankfurt.de/indexe.htm` (visited on July 12, 2019).

[8] *Under-ConScript Unicode Registry*. URL: `http://www.kreativekorp.com/ucsur/` (visited on July 20, 2019).