IP Review of Variants for the Proposed Latin LGR dated 2019-10-11

DATE: 2019-11-01

# Overview

This document reviews the variants as defined in the proposed Latin LGR (ICANN-LGP-Proposal V5 merged with Appendix-Repaired-Final.docx), and lgr-4-Latin-Script-2019-10-20-en.xml. Other aspects of the proposal will be reviewed separately.

Additional sections have been added, these are marked as “Addendum” for convenience.

# General Remarks

In analyzing in-script variants, a common case involves cross-language variants. For IDNs it is not safe to assume that labels will be in a given language; malicious registrations may well insert an unfamiliar form used in a language unfamiliar to the reader and presented in context that otherwise looks like a word in that reader’s language. With diacritics generally less prominent than the base form, that would seem to argue for wide-ranging variant sets containing all the marks found with a given base form. What argues against that is the general feature of the Latin script, which is its extensibility to other languages by adding diacritical marks. Unless there are special circumstances, users can be considered aware of this basic feature of the Latin script and expected to treat any unfamiliar marks with the same attention as applied to the distinction between similar letters, like ‘i’ and ‘j’ or ‘n’ and ‘m’. Largely we find that the proposed Latin LGR strikes the correct balance here; for items where we may disagree, see the following sections.

# Variant relationship between Base character and Base + diacritic

These variants create in principle a NULL variant situation, but this is avoided by not allowing the diacritics on their own, as is the case in any of the Latin LGR and correlated LGRs. However, it does create additional variants when the base character is part of a cross-script variant. Some of the variants introduced by transitivity might not have been considered as variants on their own. Therefore variants between Base/Base+diacritic should be minimized as much as possible.

# Variant sets introduced by underlining effects

The Latin GP should be commended for its thorough analysis of the underlining effect on variants. In that aspect, the IP agrees with some of its conclusion.

For example, the IP agrees with confusability between a base character and the same base character associated with a mark consisting of a **dot below**, **macron below**, **line below**, or **underline**. The rationale is that these marks have a single line of “ink” which can be easily confused with an underline style applied to the domain name commonly used to highlight hyperlinks.

Example: i i̱ ị (at 16 pts) i i̱ ị (at 10pts)

(The example shows the effect of underlining U+0069 LATIN SMALLER LETTER I, U+0069+U0331 LATIN SMALL LETTER I WITH MACRON BELOW, and U+1ECB Latin SMALL LETTER I WITH DOT BELOW. The confusability aspect is more pronounced at small point sizes)

However, the IP considers that the analysis may have gone too far in some aspects. The IP disagrees with creating variants for marks that are attached and look like larger and more complex descenders and are not merely made of a single point or horizontal line. Diacritics of this kind for which the Latin LGR have proposed variant sets between ‘base character’ and ‘base character with diacritics’ and which are being disputed here are:

**Cedilla**, **ogonek**, **comma below**, and **circumflex below.**

(It should be noted that the Latin LGR does not propose a variant relation between C and C WITH CEDILLA, which could be seen as not consistent with the current underline policy)

In a similar fashion, the Latin LGR has created variant sets with non-decomposable characters hat have some similarity with base characters but with integral descenders not reduced to a single dot or horizontal line. The IP does not recommend having these included in variant sets. These characters are:

þ U+00FE LATIN SMALL LETTER THORN

ŋ U+014B LATIN SMALL LETTER ENG

џ U+045F CYRILLIC SMALL LETTER DZHE

ɣ U+0263 LATIN SMALL LETTER GAMMA

The case of ỵ U+1EF5 LATIN SMALL LETTER Y WITH DOT BELOW is also peculiar because it is not rendered with a **dot below** but rather ‘dot behind and slightly low’. While an underline could possibly erase the distinction, the main reason the IP would see it as problematic is because it may be confused with ‘y.’. Because of this, its removal from the Latin LGR should be considered.

Finally, the Marshallese orthography is apparently inconsistent in using either the **dot below** or the **cedilla** (ref <https://en.wikipedia.org/wiki/Cedilla> , <https://en.wikipedia.org/wiki/Marshallese_language> , and <https://omniglot.com/writing/marshallese.php>) although the cedilla is clearly preferred. This has several consequences on the Latin LGR:

The Latin Proposal claims in section 6.1.3.4.1 page 42: “*First, certain diacritics may be considered conceptually the same as others by significant parts of the user community, such as dot below or a comma below*.” This claim appears related to Marshallese but if so, it should be addressing the confusion between **cedilla** and **dot below** instead. The **comma below** mark is only used in Romanian and it does not appear to be any confusion in the Romanian community for these marks. (Notwithstanding that some fonts may implement **cedilla below** with a comma shape).

However, even if the confusion between the **cedilla** and the **dot below** is accepted for Marshallese, it is clear the usage of the **dot below** was the result of implementation deficiency in rendering the correct sequence and used as a temporary remedy. In the view of the IP, the only two possible outcomes are either:

1. to remove both representation from the LGR (l, m, n, o with **dot below** and l, m, n, o with **cedilla**) as representing an unstable orthography,
2. to keep the preferred representation (l, m, n, o with **cedilla**) and remove l, m, n with dot below from the repertoire (o with dot below is in fact used by other LGR languages and should remain in all cases).

The IP preference would be to keep the preferred representation (option 2), and to remove the alternate representation as deprecated.

## List of variant sets reviewed

The following list enumerates the Latin LGR variant sets affected by the above consideration and suggests possible adjustments. Items marked in color are suggested to be removed from the variant set (that is, unless otherwise indicated, they should lose their variant mappings, turning them into singletons). Items struck out are suggested for possible removal from the repertoire altogether. Items in blue are suggested additions to the variant sets.

### Variant Set 1 — 6 Members (remove U+105 and make it a singleton):

a U+0061 LATIN SMALL LETTER A

a̱ U+0061+U+0331 LATIN SMALL LETTER A + COMBINING MACRON BELOW

ą U+0105 LATIN SMALL LETTER A WITH OGONEK

α U+03B1 GREEK SMALL LETTER ALPHA

а U+0430 CYRILLIC SMALL LETTER A

ạ U+1EA1 LATIN SMALL LETTER A WITH DOT BELOW

(The case for making a variant mapping between U+0061 and U+03B1 is still open)

### Variant Set 2 — 2 Members (make them singletons):

b U+0062 LATIN SMALL LETTER B

þ U+00FE LATIN SMALL LETTER THORN

### Variant Set 3 — 2 Members (see variant set 29):

### Variant set 4 – 2 Members (make them singletons)

d U+0064 LATIN SMALL LETTER D

ḓ U+1E13 LATIN SMALL LETTER D WITH CIRCUMFLEX BELOW

### Variant set 5 – 5 Members (remove U+0119 and make it a singleton)

e U+0065 LATIN SMALL LETTER E

e̱ U+0065+U+0331 LATIN SMALL LETTER E + COMBINING MACRON BELOW

ę U+0119 LATIN SMALL LETTER E WITH OGONEK

е U+0435 CYRILLIC SMALL LETTER IE

ẹ U+1EB9 LATIN SMALL LETTER E WITH DOT BELOW

### Variant set 9 – 3 Members (remove U+012F and make it a singleton)

j U+006A LATIN SMALL LETTER J

į U+012F LATIN SMALL LETTER I WITH OGONEK

ј U+0458 CYRILLIC SMALL LETTER JE

### Variant set 10 – 2 Members (make them singletons)

k U+006B LATIN SMALL LETTER K

ķ U+0137 LATIN SMALL LETTER K WITH CEDILLA

### Variant set 11 – 5 Members (remove U+013C, U+1E37, and U+1E3D from the variant set, and make U+013C and U+1E3D singleton; U+1E37 should be removed from the LGR altogether)

l U+006C LATIN SMALL LETTER L

ļ U+013C LATIN SMALL LETTER L WITH CEDILLA

ӏ U+04CF CYRILLIC SMALL LETTER PALOCHKA

~~ḷ U+1E37 LATIN SMALL LETTER L WITH DOT BELOW~~

ḽ U+1E3D LATIN SMALL LETTER L WITH CIRCUMFLEX BELOW

### Variant set 12 – 2 Members (for consistency it should have also contained U+006D+U0327, U+1E43 should be removed from the LGR altogether, the end result is that U+006D and U+006D+U+00327 should be singleton)

m U+006D LATIN SMALL LETTER M

m̧ U+006D+U+0327 LATIN SMALL LETTER M + COMBINING CEDILLA

~~ṃ U+1E43 LATIN SMALL LETTER M WITH DOT BELOW~~

### Variant set 13 – 7 Members (remove U+0146, U+014B, U+1E47 and U+1E4B from the variant set, and make U+0146, U+014B and U+1E4B singleton; U+1E47 should be removed from the LGR altogether)

n U+006E LATIN SMALL LETTER N

ņ U+0146 LATIN SMALL LETTER N WITH CEDILLA

ŋ U+014B LATIN SMALL LETTER ENG

ո U+0578 ARMENIAN SMALL LETTER VO

~~ṇ U+1E47 LATIN SMALL LETTER N WITH DOT BELOW~~

ṉ U+1E49 LATIN SMALL LETTER N WITH LINE BELOW

ṋ U+1E4B LATIN SMALL LETTER N WITH CIRCUMFLEX BELOW

### Variant set 14 – 6 Members (for consistency it should have also contained U+006F+U0327, but that should be a singleton; other script mappings to be added shown in blue)

o U+006F LATIN SMALL LETTER O

o̧ U+006F+U+0327 LATIN SMALL LETTER O + COMBINING CEDILLA

o̱ U+006F+U+0331 LATIN SMALL LETTER O + COMBINING MACRON BELOW

ο U+03BF GREEK SMALL LETTER OMICRON

о U+043E CYRILLIC SMALL LETTER O

օ U+0585 ARMENIAN SMALL LETTER OH

ଠ U+0B20 ORIYA LETTER TTHA

ഠ U+0D20 MALAYALAM LETTER TTHA

ဝ U+101D MYANMAR LETTER WA

### ọ U+1ECD LATIN SMALL LETTER O WITH DOT BELOW

(This set, when integrated to the LGR needs to pick up additional variants based on other scripts with also have a circle glyph: U+0B20, U+0D20 and U+101D. However, that process requires that at least one LGR actually lists the full set; therefore, we are requesting the Latin LGR to formally add these variants.)

### Variant set 18 – 5 Members (U+015F and U+0219 should be removed from that variant set and make into their own 2-member variant set)

s U+0073 LATIN SMALL LETTER S

ş U+015F LATIN SMALL LETTER S WITH CEDILLA

ș U+0219 LATIN SMALL LETTER S WITH COMMA BELOW

ѕ U+0455 CYRILLIC SMALL LETTER DZE

ṣ U+1E63 LATIN SMALL LETTER S WITH DOT BELOW

### Variant set 20 – 4 Members (U+021B and U+1E71 should be removed from that variant set and made singletons)

t U+0074 LATIN SMALL LETTER T

ț U+021B LATIN SMALL LETTER T WITH COMMA BELOW

ṭ U+1E6D LATIN SMALL LETTER T WITH DOT BELOW

ṱ U+1E71 LATIN SMALL LETTER T WITH CIRCUMFLEX BELOW

(note that U+0163 LATIN SMALL LETTER T WITH CEDILLA is not part of the Latin LGR)

### Variant set 21 – 7 Members (U+0173 and U+045F should be removed from that variant set and made singletons)

u U+0075 LATIN SMALL LETTER U

ų U+0173 LATIN SMALL LETTER U WITH OGONEK

ʋ U+028B LATIN SMALL LETTER V WITH HOOK[[1]](#footnote-1)

υ U+03C5 GREEK SMALL LETTER UPSILON

џ U+045F CYRILLIC SMALL LETTER DZHE

ս U+057D ARMENIAN SMALL LETTER SEH

ụ U+1EE5 LATIN SMALL LETTER U WITH DOT BELOW

### Variant set 24 – 6 Members (despite its name U+0263 is not confusable with its namesake Greek character and should be a singleton, U+1EF5 has issue of its own, but should not be part of the underlining logic and should be either removed from the LGR or made a singleton)

y U+0079 LATIN SMALL LETTER Y

ɣ U+0263 LATIN SMALL LETTER GAMMA

γ U+03B3 GREEK SMALL LETTER GAMMA

у U+0443 CYRILLIC SMALL LETTER U

ү U+04AF CYRILLIC SMALL LETTER STRAIGHT U

~~ỵ U+1EF5 LATIN SMALL LETTER Y WITH DOT BELOW~~

(This set introduces an in-script variant with Cyrillic due to inclusion of U+04AF; this should be subject of a separate discussion).

### Variant set 29 – 2 Members (for consistency with the current underlining policy, U+0063 and U+0441 should have been part of this set, but IP recommend to keep them in their own variant set (#3))

c U+0063 LATIN SMALL LETTER C

ç U+00E7 LATIN SMALL LETTER C WITH CEDILLA

с U+0441 CYRILLIC SMALL LETTER ES

ҫ U+04AB CYRILLIC SMALL LETTER ES WITH DESCENDER

# Other variant sets included in the LGR

The IP is not convinced that the all member of the two following variant sets are confusable.

### Variant set 43 – 3 Members (U+1EA7 looks different from the others and could be a singleton)

ă U+0103 LATIN SMALL LETTER A WITH BREVE

ӑ U+04D1 CYRILLIC SMALL LETTER A WITH BREVE

ầ U+1EA7 LATIN SMALL LETTER A WITH CIRCUMFLEX AND GRAVE

### Variant set 47 – 3 Members (U+03C3 looks quite different from the others and could be a singleton)

ơ U+01A1 LATIN SMALL LETTER O WITH HORN

σ U+03C3 GREEK SMALL LETTER SIGMA

ợ U+1EE3 LATIN SMALL LETTER O WITH HORN AND DOT BELOW

## Addendum: Possibly mistaken reflexive mapping

The code point U-028B ʋ LATIN SMALL LETTER U WITH HOOK has a reflexive mapping to “out-of-repertoire-var” in the latest XML. This would remove that code point from the repertoire of the Latin LGR. If this was unintentional, the reflexive mapping needs to be removed, otherwise, the whole code point can go. “Out-of-repertoire-var” mappings that go to code points not in *any other* LGR (present or forthcoming) are not meaningful.

## Addendum: Possibly missing variant

Appendix D contains a rationale for making U+0066 f a variant of U+0192 ƒ LATIN SMALL LETTER F WITH HOOK, and notes that it was decided by the GP to make these variants. But neither section 6.5 nor the XML follow suit. This appears to be an inconsistency that needs to be resolved one way or another; either by a fix of the Appendix to indicate why these were not made variants after or, or by a fix to the formal definitions.

NOTE: this variant set is not covered by the review above because that review is based on Section 6.5 and the XML. Therefore, there is no IP position on it at this time.

# Allocatable variants

We note that the spelling variation between “ss” and “ß“ between standard Swiss and other German-language orthographies has been suggested as allocatable in the proposal by the Latin GP by adding an allocatable variant from ‘ß’ to ‘ss’ (but not in the reverse direction). This appears based on the fact that Swiss users normally use only the “ss” form with no “ß“ on their keyboards while not all German words spelled with “ss” have a possible equivalent spelling with “ß“. In the proposal, there is no formal restriction on the number of possible allocatable variant labels for any label containing “ß“ although there may be an implicit limit on the number of “ß“ in any labels other than some rather fanciful ones.

 However, it seems sensible to adopt a restriction requiring that either all “ß“ are mapped to “ss” or none. This cannot be addressed by a “no-mix” rule in the style used in the Arabic LGR, because “ss” and “ß“ can and do coexist in German words, but it could be achieved by using a subtype for the variant type with a corresponding <action> defined that resolves only labels where all mappings are taken as “allocatable”.

## Addendum: Minimizing allocatable variants

As defined the variant set can lead to **multiple allocatable variants**. With German labels (especially with German compound nouns) there can easily be multiple "ß". A non-contrived example is "Außenmaß" (outer dimension). As a label, this would have 4 allocatable variants, but only two are needed (the Swiss would write all "ß" with "ss", not just the first or the last). However, it is possible to mix "ss" and "ß" in German, as in "Außeninteresse" (outside interest). That label would have only one allocatable variant, which is fine; however, the alternation between "ss" and "ß" makes it impossible to define a "no-mix" rule between "ss" and "ß".

The situation of multiple orhographies sharing a code space looks a bit like Chinese variants, where simplified and traditional ways of writing coexist with those that are used in both. In that scenario, various forms of “subtyping” of the “allocatable” variant type are used. For a generic discussion see for example RFF 8228.

What if we made "ß" a reflexive "r-german" and gave the mapping "ß" --> "ss" the type "swiss"?

Then we could define these actions:

<action disp="allocatable" all-variants="swiss" />

<action disp="allocatable" all-variants="r-german" />

Any mixed variant label that contains both an original "ß" but also some that have been mapped to "ss" would fail the "all-variants" test, but one that either has all the original "ß" or only those mapped to "ss" would pass. (Any original "ss" wouldn't have a typed variant at all and therefore be processed normally).

For those not familiar with reflexive variants: these map the code point to itself and are “applied” to the original label, or any code position that that is unchanged in a given variant label permutation.

(There are some edge cases that this scheme may not support, such as any word spelled with two "ß" in the old orthography, but only one in the modern one: you wouldn't be able to support both old and modern orthography in that case - in the view of the IP, the need to reduce multiple allocatable variants would tend to outweigh that level of linguistic fidelity - and the number of non-contrived examples should be rather low in any case: dual "ß" are not that frequent to begin with).

Finally, let's check a triple "sss" to see whether this would get into any trouble:

ѕѕѕ --> sss ->  --> sß  / ßs --> ṣṣṣ --> ѕѕѕ

It looks like both partitions {s}{ss} and {ss}{s} are well-behaved. Because we only had an overlap, not a null-variant, the sequences cannot get longer. As long as we handle the overlap by adding all the other variants for "ss" into the variant set with "ß" as suggested above, the resulting variant set should be well-behaved.

NOTE: it is true that "sß"  is an impossible combination in a German word (since "ß" has to follow a vowel, so "ßß" is likewise impossible). Should such combination be explicitly banned in labels, or is non-existence in German text to be considered a mere spelling rule?

To answer this, consider “xkcd” which is not a word in any language written in the Latin script and yet, there's <https://xkcd.com/>.

In the Latin script, whether these can occur as words or not is based on spelling rules which are not enforced in the LGR. (Other than for combining marks, all combinations go).

## Addendum: Overlapped variants

There is an **overlapped** v**ariant** between the variants defined for “ss” and “s”. (See RZ-LGR-3 section 6.6 for a general discussion). In the Latin LGR, there is a cross-script variant for s (and the Latin GP also maps it to some combinations of s+diacritic as well). That "s" overlaps with the sequence "ss". As a result, the "ss" in a label like "pass" (passport) would have some variants which themselves would not have a "ß" as a variant. Therefore, transitivity among variant labels is broken.[[2]](#footnote-2)

E.g. paṣṣ ---/---> paß ---> pass --->  paṣṣ and paṣṣ <---/-- paß <--- pass <---  paṣṣ

(In this example "paß " isn't a spelling that is currently correct, but until recently, in the old pre-1996 orthography that many users are still familiar with, it used to be the standard one.)

The same applies to the cross-script variant to the Cyrillic Dze (U+0455) which looks like "ѕ":

раѕѕ <---/---> paß <---> pass <---> раѕѕ

In order to make the variant label set well-behaved, the solution is to add all the other variants for "ss" into the variant set with "ß". Those variants would be derived from mapping “ss” to all possible combinations of pairs of variants for “s” (including mixed pairs).

# Other characters not included in LGR variant sets

Other characters part of the Latin LGR have confusability issues that have not been explored in the proposal. These are:

ď U+010F LATIN SMALL LETTER D WITH CARON

ľ U+013E LATIN SMALL LETTER L WITH CARON

Check for example the French word ‘aujourd’hui’ (correct spelling with an apostrophe or better a smart quote) ‘aujourďhui’ (using U+010F); or ‘l’ornière’ (correct spelling), versus ‘ľornière’ (using U+013E). The confusion is more pronounced with serif fonts than with sans serif fonts, but it still is a concern that should be addressed by the Latin GP, including possible removal of these two characters from the LGR repertoire.

Such a restriction might seem, at first blush, as too limiting; however, it should be considered in the context of a similar exclusion for 02BC MODIFIER LETTER APOSTROPHE.

1. See section 5.2 [↑](#footnote-ref-1)
2. The ability to shortcut the match for blocked variants depends on well-behaved sets for variant labels. As also discussed in RFC 8228, having well-behaved sets of code point variants is not by itself a guarantee for well-behaved sets of variant labels if sequences are involved. (Additional guidance can be found in RZ-LGR-3 Section 6). [↑](#footnote-ref-2)