IP Response to Malayalam Variant Query

DATE: 2018-12-21

# Overview

This document replies to a query forwarded on behalf of Malayalam sub-group of the Neo-Brahmi GP regarding possible issues with the WLE rules as affecting U+0D31 and U+0D33.

# The Query

On 12/11/2018 12:28 PM, Sarmad Hussain wrote:

Dear IP members,

We have received a query from Malayalam sub-group of NBGP about the current rules not being able to address some common labels – where X-Halant-X is only allowed in one context but is needed in both contexts:

                X  X-Halant-X

                X-Halant-X  X

Based on public comment, there are two possible code points X now: 0D33 and 0D31 (this added because in some systems the stacked version is not rendered and it is rendered side-by-side).  See the examples below.

 മീറ്ററ്  - Metre – Metre  (X-Halant-X  X) -  X-Halant-X = /t/; X = /r/

മിററ് -Mirror – Mirror  (X X Halant)

മീററ്റ് -Meerut – Meerut (X  X-Halant-X)

ഉള്ളളവ് - Ullalavu - A unit of volume (X-Halant-X  X) -  X-Halant-X = /ll/; X = /l/

നീളള്ള -neelalla – lengthy (X  X-Halant-X)

ഒള്ളള - Ollala - It's a place name (X-Halant-X  X)

Please note that 0D31 is not sub-joined on some platforms and appears on the baseline.

 Please advise how to address this in the XML for Malayalam.

 Regards,
Sarmad

# Background

Originally, the GP had desired to make the pair ള്ള vs ളള variant sequences of each other. The IP had noted that there are certain issues with that approach, due to the variant being an “effective null variant” (see below) and had attempted to find a solution using WLE rules and sequences, thereby sidestepping those issues. The discussion was based on the input from the GP as to what combinations might be expected to occur.

The query points out that additional combinations may have to be taken into account and when the IP reviewed the public comments on Malayalam it found that one commenter did find fault with the level of analysis in the LGR draft and also suggests that the issue is more complex than we anticipated in other ways. The good news is that the comment agrees "the ള്ള vs ളള pair is indeed worth discussing, since this pair is probably the single most confusable pair in the reformed orthography".

The IP investigated a possible solution based on refining the WLE rules, but ran into some unsatisfactory complexities. Therefore, this document approaches the issue from a fresh perspective.

# Analysis

Beyond the generic “effective null variant” problem (below) the public comments raised another issue with the attempt to define ള്ള vs ളള as variant sequences. (In the following analysis, we substitute X for U+0D33, as much of the analysis and solution holds for both U+0D33 and u+0D31.)

The comment noted that the **code point** sequence X X R where X is 0D33 and R is a reordrant vowel sign (such as VOWEL SIGN E) will result in the **glyph** sequence: X R X, with the glyph for the vowel sign separating the two consonants. This sidesteps the issue that we are trying to solve, i.e. that X Halant X looks like X X.

 (ളള + െ → ളളെ) X - X – R
 (ള്ള + െ → ള്ളെ) X - halant - X R

Repetition of above as an image with the reordrant vowel highlighted:

 

Because of the way a reordrant vowel interferes with the look-alike presentation, we cannot write any rules without also checking reordrant vowels that follow (in this example: 0D46)

This requires us to analyze this further to be sure that the recommendation we give the GP is the most appropriate.

Liang further suggests that disallowing ളള could be problematic, because of the way it could occur when words are written together without space. As labels do not allow a space and nothing prevents words from written together, this may indeed be an issue. If we can arrive at a solution that does not require to disallow this sequence, at this point, it may be preferable.

## Possible options:

We may have to go back to the drawing board. The issue started with a desire to have ള്ള vs ളള pair as variants. This would normally not be an issue, except that it creates an “effective null variant for Halant. In the mapping

 ( X - Halant - X <—>.X X )

the Halant on the left corresponds to a “null” position on the right. These “effective null variants” are not well-behaved; without additional safeguards they can lead to an effect akin to recursion.

We can reign in the "recursion" aspect of that variant, by adding a context rule to the variant definition:

 ( X - Halant - X <—>.X X )  not-when="followed-by-X"

That context restriction prevents the pattern from growing with each iteration.

Per Liang's analysis, we would also need a new context rule:

 not-when="followed-by-R"

where R is any reordrant vowel, because once the sequences are followed by a reordrant vowel sign they no longer look the same. These two contexts are both RHS contexts, so they can be merged via <choice>, thus not violating the restriction on only having single contexts.

The effect of the context rules on the variant mapping is to remove the variant mapping when the context is not satisfied. In contrast, the result of any context rules on a code point or code point sequence is to disallow labels with that code point or sequence in that context. Note that the restriction on a code point does not apply if that code point is part of a sequence that is enumerated separately, even if it contains a sub-sequence and its disallowed context.

This ability to use code point contexts allows us to write variant definitions that are specific to a particular number of X, but forces us to enumerate a small number of sequences.

## Putting it all together

First, we need a context rule to disallow X X unless part of an enumerated sequence. By defining

 X : X not-when="followed-by-X"

We disallow any X from following another X unless they are part of an enumerated XX sequence.

The X here can stand in principle for both 0D31 and 0D33, but when X is substituted, each would generate a specific context rule.

Now we are ready to enumerate the XX sequence and its variant:[[1]](#footnote-1)

 XX not-when="followed-by-X" (to limit longer sequences to enumerated ones)
     variant of: X - Halant - X, not-when="followed-by-X-or-R" (to limit the variant)

 X-Halant-X not-when="followed-by-X" (to limit longer sequences to enumerated ones)
     variant of: X X, not-when="followed-by-X-or-R"

Again, we limit longer sequences to enumerated ones. We have a combined context rule on the variant mapping that prevents it from being applied if a reordrant vowel follows (and reigns in any “recursion”).

Finally we would need to enumerate the "triple X with Halant" case:

 X X Halant X not-when="followed-by-X" (to limit longer sequences to enumerated ones)
     variant of: X - Halant - XX, not-when="followed-by-X-or-R"

 X-Halant-X X not-when="followed-by-X" (to limit longer sequences to enumerated ones)
     variant of: X X -halant- X , not-when="followed-by-X-or-R"

Note: Having the context not-when="followed-by-X" defined on ***both*** code point sequence and variant is something that the IP has worked out as "best practice" for dealing with null-variants (to prevent "recursion").

### Optional Sequence

The GP would need to decide whether “X X X” can be expected to occur - including the case of two words meeting. If the sequence can be left out from the LGR, it would have the effect of disallowing it. However, if that sequence is needed, it can be added (with sequence context of not-when=”followed-by-0D33”). It would also need to be a variant of both the other sequences X X Halant X and X Halant X , and variant mappings would have to be added (with variant context: not-when="followed-by-X-or-R").

 X X X, not-when="followed-by-X"

     variant of: X - Halant – X X, not-when="followed-by-X-or-R"

    variant of: X X -halant- X , not-when="followed-by-X-or-R"

 X X Halant X not-when="followed-by-X" (to limit longer sequences to enumerated ones)
     variant of: X - Halant – X X, not-when="followed-by-X-or-R"

     variant of: X X X, not-when="followed-by-X-or-R"

 X-Halant-X X not-when="followed-by-X" (to limit longer sequences to enumerated ones)
     variant of: X X -halant- X , not-when="followed-by-X-or-R"

     variant of: X X X , not-when="followed-by-X-or-R"

### Undesirable Variant

The sequence X - Halant - X - Halant - X makes one of the Halants visible and should not be a variant:

ള്ള്ള

Because X X is a part of the longer sequences X X H X and X H X X, we do need to further restrict variants to and from X X by changing the context rule on those mappings to: not-when=”follows-0D33-04D-or-followed-by-0D33-or-0D4D-0D33-or-R”. It we did not do that, then X X H X and X H X X both would be able to produce the undesirable variant X - Halant - X - Halant – X.

For X X X X and any sequences it containing four X with one or more Halant interspersed we would simply not enumerate them and therefore disallowing such sequences.

### Differences between 0D31 and 0D33

If any of the generic (X-based) sequences cannot occur or are not variants or each other for U+0D31 they don’t have to be enumerated. The IP leaves it to the GP to define which sequences occur and which need to be variants of each other (given the rendering infidelities alluded to in the query).

## XML Fragment

The following is an XML fragment showing only the sequences and rules for U+0D33; they can be duplicated for U+0D31 as needed.

In this fragment, no comments, tags or references are shown. They would need to be supplied for the actual XML. Also, all reordrant vowels (like 0D46) would need to be given the tag “R”.

<char cp=”0D33” not-when=”followed-by-0D33” />

<char cp=”0D33 0D33” not-when=”followed-by-0D33” >

 <var cp=”0D33 **0D4D** 0D33” not-when=”followed-by-0D33-or-R” />

</char>

<char cp=”0D33 **0D4D** 0D33” not-when=”followed-by-0D33” >

 <var cp=”0D33 0D33” not-when=”followed-by-0D33-or-R” />

</char>

<char cp=”0D33 0D33 **0D4D** 0D33” not-when=”followed-by-0D33” >

 <var cp=”0D33 **0D4D** 0D33 0D33” not-when=”followed-by-0D33-or-R” />

</char>

<char cp=”0D33 **0D4D** 0D33 0D33” not-when=”followed-by-0D33” >

 <var cp=”0D33 0D33 **0D4D** 0D33” not-when=”followed-by-0D33-or-R” />

</char>

<class from-tag=”R” />

<rule name= “followed-by-0D33”>

 <anchor/>

 <look-ahead>

 <class by-ref=”R” />

 </look-ahead>

</rule>

<rule name= “followed-by-0D33-or-R”>

 <anchor/>

 <look-ahead>

 <choice>

 <class by-ref=”R” />

 <char cp=”0D33” />

 </choice>

 </look-ahead>

</rule>

<rule name= “folllows-0D33-0D4D-or-followed-by-0D33-or-0D4D-0D33-or-R”>

 <choice>

 <rule>

 <look-behind>

 <char cp=”0D33 0D4D” />
 </look-behind>
 <anchor/>

 </rule>

 <rule>

 <anchor/>

 <look-ahead>

 <char cp=”0D4D 0D433” />
 </look-ahead>
 </rule>

 <rule by-ref=” followed-by-0D33-or-R” />

 </choice>

</rule>

1. (actual context rule needs to be: not-when=”follows-0D33-04D-or-followed-by-0D33-or-0D4D”, see below) [↑](#footnote-ref-1)