

Development of OpenType Tai Tham font: NS Tahlom 3

Introduction	2
Logical input order	3
Logical input order: Example for complex vowel forms	4
Standrad ligature (liga) & Glyph composition (ccmp)	5
LIGA 1 - Base ligature	5
L1.1 nya-nya and na-nya ligature [uni1A31 + uni1A2C, uni1A2C + uni1A2C]	5
L1.2 dura ligature [uni1A2F + uni1A6A + uni1A55]	5
L1.3 Ila-ha ligature [uni1A4A + uni1A49]	5
L1.4 Punctuation Danda, Satkaan [uni1A48, uni1AAA]	6
L1.5 Naa and Nam ligature [uni1A36 + uni1A63 + uni1A74] section 1 (no LookupFlag)	6
L1.6 Sara-am ligature	6
L1.7 Maisam with long aa [uni1A7B, uni1A64]	7
LIGA 2 - Abovebase ligature	7
L2.1 Mai sat & Tone ligature [uni1A62 + uni1A75 / uni1A76]	7
L2.2 Pali sara-i & nikkahit ligature [uni1A65 + uni1A74]	8
LIGA 3 - Ligatures with LookupFlag	8
L3.1 Uy-Ub ligature [uni1A69 + uni1A37 / uni1A3F]	8
L3.2 Letter with haag pa [uni1A5B]	9
L3.3 Belowbase form and postbase form of the letters CCMP vs LIGA	9
L3.4 Naa ligature [uni1A36 + uni1A63] section 2 (with LookupFlag)	11
Pre-Base form (pref)	12
Contextual alternate (calt)	13
CALT 1 - Removal of uni25CC dot (automatically added)	13
C1.1 Non-spacing glyph and postbase glyph	13
C1.2 Spacing vowel sara-a	13
C1.3 Consonant signs	14
C1.4 Prebase glyphs	14
CALT 2 - Base substitution (shorten haang)	14
C2.1 Haang-group substitution	14
C2.2 Special case of Nya [uni1A2C]	16
CALT 3 - Belowbase substitution (optional)	17
C3.1 Belowbase na-ma substitution	17
C3.2 Second stack substitution: wa [uni1A45]	17
C3.3 Second stack substitution: sara-u [uni1A69, uni1A6A]	18
CALT 4 - Postbase substitution	19
C4.1 Postbase Ya [uni1A3F]	19
C4.1 Postbase Ba [uni1A37, uni1A38]	19
Positioning (GPOS)	21
POS 1 - Glyph class for mark positioning	21
POS 2 - General mark-to-base positioning	21
POS 3 - Sara-am positioning	21
POS 4 - Mai Sam positioning	21
POS 5 - Kern for numerals	21
Limitations	22

Introduction

This document is intended to be a record for the development of the OpenType features of Tai Tham font NS Tahlom version 3.024. The current version of the font is developed using Glyphs 3.0 and is intended mainly for MacOS system and Google Chrome web browser.

The input order here follows and is adapted from:

1. The document L2/07-007R « <https://www.unicode.org/L2/L2007/07007r-n3207r-lanna.pdf> » proposed by Unicode consortium in 2007.
2. The document L2/19-365 « <http://www.unicode.org/L2/L2019/19365-tai-tham-structure.pdf> »

Logical input order

Order	Input order	Example	Note
Base letter	င	င	
Medial or Initial consonant cluster (either belowbase or postbase) (ccmp feature)	င ဖ င ဖ ဃ င ဖ ဃ	င (င) င (င) င (င)	Medial characters ဖ ဖ Hor-nam form ဃ ဃ ဃ ဃ ဃ ဃ And other semi-syllable cluster
Rawong	င ဖ	င	
Prebase vowels	င ဖ ဃ င ဖ ဃ င ဖ ဃ ဃ	င င င	င ဃ ဃ ဃ ဃ ဃ ဃ
Belowbase vowels	င ဖ င ဖ င ဖ ဃ	င င င	င ဃ ဃ ဃ
Abovebase vowels	င ဖ ဃ င ဖ ဃ င ဖ ဃ ဃ ဃ	င င င	င ဃ ဃ ဃ ဃ ဃ ဃ
Tone marks	င ဖ ဃ င ဖ ဃ ဃ	င င	င ဃ ဃ ဃ
Mai-Sam	င ဖ ဃ ဃ င ဖ ဃ ဃ ဃ	င င	Due to limitation with current shaping, Mai-sam can only be placed here, after finishing all abovebase sings.
Belowbase final consonant (sakot, liga feature)	င ဖ ဃ	င	
Postbase final consonant (sakot)	င ဖ ဃ	င	
Spacing vowels (postbase vowels)	င ဖ ဃ င ဖ ဃ	င င	
Sara-Am	င ဖ ဃ င ဖ ဃ င ဖ ဃ င ဖ ဃ င ဖ ဃ	င င င င င	
Raham / Karant	င ဖ ဃ င ဖ ဃ	င င	
Mai-Sam as word repeating sign	င ဖ ဃ	င	

Logical input order: Example for complex vowel forms

[illegible]

Standrad ligature (liga) & Glyph composition (ccmp)

A longer argument comes before a shorter one, and LookupFlag comes the last. Therefore in this section, the lookups are placed in the ordered not by its importance but by its technicality.

LIGA 1 - Base ligature

L1.1 nya-nya and na-nya ligature [uni1A31 + uni1A2C, uni1A2C + uni1A2C]

Nya-nya ligature is the required ligature that cannot be omitted. It forms between ဃ uni1A31 + ဃ uni1A2C or ဃ uni1A2C + ဃ uni1A2C.

```
feature liga {
lookup nya_nya_liga {
    sub uni1A31 uni1A60 uni1A2C by uni1A2C.1A2C;
    sub uni1A2C uni1A60 uni1A2C by uni1A2C.1A2C;
} nya_nya_liga;
} liga;
```

Testing words: ဃ ဃ ဃ ဃ ဃ

L1.2 dura ligature [uni1A2F + uni1A6A + uni1A55]

Dura ligature is created to fix the problem that the prebase glyph uni1A55 ဃ cannot function after the belowbase such as sara-u [uni1A6A]. So that we cannot type uni1A2F uni1A6A uni1A55 ဃ ဃ ဃ and expect it to give us ဃ ဃ ဃ.

```
feature liga{
lookup dura_liga {
    sub uni1A2F uni1A6A uni1A55 by uni1A2F.1A6A.1A55;
} dura_liga;
} liga;
```

Testing words: ဃ

L1.3 lla-ha ligature [uni1A4A + uni1A49]

This ligature is rather optional. Its occurrence is really limited but important for the often used Pali word ဃဃဃဃဃ.

```
feature liga {
lookup lla_ha_liga {
    sub uni1A4A uni1A60 uni1A49 by uni1A4A.1A49;
} lla_ha_liga;
} liga;
```

Testing words: ဃဃဃဃဃ ဃ

L1.4 Punctuation Danda, Satkaan [uni1A48, uni1AAA]

For the ease while typing. Reducing keyboard keys, and for the correct encoding.

```
lookup punctua_danda {
    sub uni1AA8 uni1AA8 by uni1AA9;
    sub uni1AAA uni1AAA by uni1AAB;
} punctua_danda;
feature liga{ lookup punctua_danda;};
```

Testing words: သ

L1.5 Naa and Nam ligature [uni1A36 + uni1A63 + uni1A74] section 1 (no LookupFlag)

Sara-am has the input order for ၵ = ၵ ၵ ၵ and ၵ = ၵ ၵ ၵ . For chrome/web rendering, uni25CC dot is always added after tone marks. So, don't forget to add uni25CC in the argument. For MacOS rendering, adding uni25CC isn't needed but the Nam ligature with tone marks ၵ ၵ are required and must be placed before the whole Sara-am ligature.

```
lookup liga_naa_ext {

# with 25CC removal, for web/chrome rendering
sub uni1A36' uni1A75' uni25CC' uni1A63' uni1A74' by uni1A36.1A75.1A63.1A74;
sub uni1A36' uni1A76' uni25CC' uni1A63' uni1A74' by uni1A36.1A76.1A63.1A74;
sub uni1A36' uni1A63' uni25CC' uni1A74' by uni1A36.1A63.1A74;
sub uni1A36' uni1A75' uni25CC' uni1A63' by uni1A36.1A75.1A63;
sub uni1A36' uni1A76' uni25CC' uni1A63' by uni1A36.1A76.1A63;

# for MacOS rendering
sub uni1A36' uni1A75' uni1A63' uni1A74' by uni1A36.1A75.1A63.1A74;
sub uni1A36' uni1A76' uni1A63' uni1A74' by uni1A36.1A76.1A63.1A74;

} liga_naa_ext;

feature liga{ lookup liga_naa_ext;};
feature ccmp{ lookup liga_naa_ext;};
```

Testing words for web/chrome: ၵ ၵ ၵ ၵ ၵ

Testing words for MacOS: ၵ ၵ

L1.6 Sara-am ligature

Sara-am has the input order for ၵ = ၵ ၵ ၵ and ၵ = ၵ ၵ ၵ . Therefore it makes this ligature the most annoying one and causes headache the most. Also, don't forget to add ignore sub for ၵ because this ligature can't be placed before this lookup as it should be (due to the fact that it needs LookupFlags) . This ligature will be placed in L3.4 that comes the last.

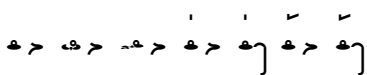
```

lookup sara-am {
# ignore sub to reserve this for naa ligature section 2 (with LookupFlag) below
  ignore sub uni1A36' uni1A63' uni1A74';
# sara am
  sub uni1A63' uni1A74' by uni1A63.1A74;
  sub uni1A64' uni1A74' by uni1A64.1A74;

# tone-1 sara-am
  sub uni1A75' uni1A63' uni1A74' by uni1A63.1A74.1A75;
  sub uni1A75' uni1A64' uni1A74' by uni1A64.1A74.1A75;
# tone-2 sara-am
  sub uni1A76' uni1A63' uni1A74' by uni1A63.1A74.1A76;
  sub uni1A76' uni1A64' uni1A74' by uni1A64.1A74.1A76;
} sara-am;

feature liga{ lookup sara-am;};

```

Testing words: 

L1.7 Maisam with long aa [uni1A7B, uni1A64]

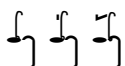
For stylistic aspect.

```

lookup maisam_aa {
# mai sam
  sub uni1A7B uni1A64 by uni1A64.1A7B;
# tone-1 mai sam
  sub uni1A75 uni1A7B uni1A64 by uni1A64.1A75.1A7B;
# tone-2 mai sam
  sub uni1A76 uni1A7B uni1A64 by uni1A64.1A76.1A7B;
} maisam_aa ;

feature liga{ lookup maisam_aa;};

```

Testing words: 

LIGA 2 - Abovebase ligature

L2.1 Mai sat & Tone ligature [uni1A62 + uni1A75 / uni1A76]

This is a minor ligature and depends on the style. However, after reviewing the research on the history of Tai Tham script, I find that this ligature has more occurrence than the pure combination of Mai Sat and Tone-2. For another ligature between Mai Sat and Tone-1, it's easier this way than by mark-to-mark positioning.

```

lookup tone_mark {
  sub uni1A62 uni1A75 by uni1A62.1A75;
  sub uni1A62 uni1A76 by uni1A62.1A76;
} tone_mark;
feature liga{ lookup tone_mark;};
feature ccmp{ lookup liga_naa_ext;};

```

Testing words: နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ

L2.2 Pali sara-i & nikkahit ligature [uni1A65 + uni1A74]

This is another minor ligature but very important for Pali text as Sara-i sign [uni1A65] will merge with Nikkahit [uni1A74] to be in the same shape as Sara-ue [uni1A67]

```
lookup pali_ing_liga {
    sub uni1A65 uni1A74 by uni1A67;
} pali_ing_liga;
feature liga{ lookup pali_ing_liga;}
```

Testing words: နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ

LIGA 3 - Ligatures with LookupFlag

L3.1 Uy-Ub ligature [uni1A69 + uni1A37 / uni1A3F]

Sara-u with belowbase ba uni1A37.blw နိဗ္ဗာနံ is not considered canonical. Normally it's written as နိဗ္ဗာနံ while sara-u with belowbase ya uni1A3F in a ligature form နိဗ္ဗာနံ is considered by many as a rule. For the correct rendering, don't forget to add uni25CC in the argument as it can be inserted automatically.

```
lookup uyub_liga {

lookupflag UseMarkFilteringSet @belowbase;

# ba
sub uni1A69 uni25CC uni1A60 [uni1A37 uni1A38] by uni1A37.blw.1A69; #
sub uni1A6A uni25CC uni1A60 [uni1A37 uni1A38] by uni1A37.blw.1A6A; #
sub uni1A69 uni1A60 [uni1A37 uni1A38] by uni1A37.blw.1A69; #
sub uni1A6A uni1A60 [uni1A37 uni1A38] by uni1A37.blw.1A6A; #

# ya
sub uni1A69 uni25CC uni1A60 uni1A3F by uni1A3F.blw.1A69;
sub uni1A6A uni25CC uni1A60 uni1A3F by uni1A3F.blw.1A6A;
sub uni1A69 uni1A60 uni1A3F by uni1A3F.blw.1A69;
sub uni1A6A uni1A60 uni1A3F by uni1A3F.blw.1A6A;
} uyub_liga;

feature liga{ lookup uyub_liga;};
feature ccmp{ lookup uyub_liga;};
```

Testing words: နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ နိဗ္ဗာနံ

by the prebase glyph and will wrongly render the glyph cluster somehow like ຸ ຸ ຸ ຸ ຸ instead. This is the reason why CCMP is for the initial consonant belowbase/cluster.

However, CCMP does not work if it's placed after abovebase glyphs or vowel sign sara-aa. The word like ຸ (input ຸ ຸ ຸ ຸ) will be wrongly rendered somehow like ຸ ຸ instead if only CCMP being employed. This problem can be fixed by using LIGA (or RLIG). That's why LIGA is for the formation of final consonant belowbase/cluster.

Besides, LookupFlag dose not work with CCMP, that's also another reason why we need LIGA feature for naa ligature.

The code is rather simple but we have to tediously assign it for every individual letter. There are some exceptions or notable cases.

- 1) The belowbase form of ຸ ຸ are the same as ຸ ຸ.
- 2) The belowbase form of uni1A40 ຸ is in fact sign oy uni1A6D ຸ. Thus the belowbase substitution of uni1A60 + uni1A40 is omitted.
- 3) The belowbase form of uni1A4B ຸ is in fact the sign o below uni1A6C ຸ. Thus the belowbase substitution of uni1A60 + uni1A4B is omitted. And I suggest against using uni1A60 + uni1A4B, because it could lead to wrong encoding. It's recommended to type uni1A6C directly.

```
lookup below_char {
lookupflag UseMarkFilteringSet @belowbase;
# wak ka
sub uni1A60 uni1A20 by uni1A20.blw;
sub uni1A60 uni1A21 by uni1A21.blw;
sub uni1A60 uni1A22 by uni1A22.blw;
sub uni1A60 uni1A23 by uni1A23.blw;
sub uni1A60 uni1A24 by uni1A24.blw;
sub uni1A60 uni1A25 by uni1A25.blw;
sub uni1A60 uni1A26 by uni1A26.blw;

# wak cha
sub uni1A60 uni1A27 by uni1A27.blw;
sub uni1A60 uni1A28 by uni1A28.blw;
sub uni1A60 uni1A29 by uni1A29.blw;
sub uni1A60 uni1A2A by uni1A2A.blw;
sub uni1A60 uni1A2B by uni1A2B.blw;
sub uni1A60 uni1A2C by uni1A2C.blw;

# wak rata
sub uni1A60 uni1A2D by uni1A2D.blw;
sub uni1A60 uni1A2E by uni1A2E.blw;
sub uni1A60 uni1A2F by uni1A2F.blw;
sub uni1A60 uni1A30 by uni1A30.blw;
sub uni1A60 uni1A31 by uni1A31.blw;

# wak ta
sub uni1A60 uni1A32 by uni1A32.blw;
sub uni1A60 uni1A33 by uni1A33.blw;
sub uni1A60 uni1A34 by uni1A34.blw;
sub uni1A60 uni1A35 by uni1A35.blw;
sub uni1A60 uni1A36 by uni1A36.blw;
```

```
# wak pa
sub unilA60 unilA37 by unilA37.blw;
sub unilA60 unilA38 by unilA38.blw;
sub unilA60 unilA39 by unilA39.blw;
sub unilA60 unilA3A by unilA3A.blw;
sub unilA60 unilA3B by unilA3B.blw;
sub unilA60 unilA3C by unilA3C.blw;
sub unilA60 unilA3D by unilA3D.blw;
sub unilA60 unilA3E by unilA3E.blw;


# awak
sub unilA60 unilA3F by unilA3F.blw;
sub unilA60 unilA40 by unilA40.blw;
sub unilA60 unilA41 by unilA41.blw;
sub unilA60 unilA42 by unilA42.blw;
sub unilA60 unilA43 by unilA43.blw;
sub unilA60 unilA44 by unilA44.blw;
sub unilA60 unilA45 by unilA45.blw;
sub unilA60 unilA46 by unilA46.blw;
sub unilA60 unilA47 by unilA47.blw;
sub unilA60 unilA48 by unilA48.blw;
sub unilA60 unilA49 by unilA49.blw;
sub unilA60 unilA4A by unilA4A.blw;
# sub unilA60 unilA4B by unilA6C; # sing ow below
sub unilA60 unilA4C by unilA4C.blw;


# sara loi (non standard)
sub unilA60 unilA4D by unilA4D.blw;
sub unilA60 unilA4E by unilA4E.blw;
sub unilA60 unilA4F by unilA4F.blw;
sub unilA60 unilA50 by unilA50.blw;
sub unilA60 unilA51 by unilA51.blw;
sub unilA60 unilA53 by unilA53.blw;
sub unilA60 unilA54 by unilA54.blw;

} below_char;

# determine the features

feature liga { lookup below_char; } liga;
feature ccmp { lookup below char; } ccmp;
```

Testing words: 

It should not work with: 

L3.4 Naa ligature [uni1A36 + uni1A63] section 2 (with LookupFlag)

This ligature requires `LookupFlag` for the ligature to ignore the mark (abovebase, belowbase, postbase). This should be declared at the last lookup to prevent the `LookupFlag` interfering with other lookups that come after. This ligature also does not need to be declared in `CCMP` as the `LookupFlag` does not work in `CCMP`.

Testing words: ၃ ၄ ၅ ၆ ၇ ၈ ၉ ၁၀ ၁၁ ၁၂ ၁၃ ၁၄ ၁၅ ၁၆ ၁၇ ၁၈ ၁၉ ၂၀ ၂၁ ၂၂

Draft version 20/08/2021

Contextual alternate (calt)

Contextual alternate feature CALT (or other substitution features such as ABVS BLWS PSTS ...) must be placed after LIGA or CCMP features. We must finish building ligatures or composed glyphs before changing (substituting) glyphs with their alternate forms because sometimes we need to substitute the ligature with its alternate form too. Actually, CALT should be placed as the last feature on the GSUB table. CALT can be used globally, there is no need to use ABVS, BLWS, PSTS, etc. at this moment (2021).

CALT 1 - Removal of uni25CC dot (automatically added)

This category precedes the rest of the contextual alternate commands. 25CC dot must be removed at the beginning prior to every other CALT lookups, to prevent it being automatically added and disrupting other contextual substitution command chain in the lookups that come after.

C1.1 Non-spacing glyph and postbase glyph

25CC dot is automatically added for all non-spacing mark (or whatever glyph) i.e. abovebase, belowbase, when there is no base glyph for the mark to attach with. The problem are:

- 1) When we have 2 mark glyphs together e.g. sara-i + tone-1 [uni1A65 uni1A75], the 25CC dot is automatically added before tone-1 (uni1A75) and become [uni1A65 **uni25CC** uni1A75].
- 2) When we place those non-spacing marks on top of the postbase glyph (spacing glyph), the shaping engine does not recognize the postbase glyph as a base character, thus the uni25CC dot is automatically added, e.g. la-postbase + sara-i [uni1A60 uni1A43 uni1A65] becomes la-postbase dot-25CC sara-i [uni1A60 uni1A43 **uni25CC** uni1A65].

Thus, the 25CC dot can be deleted by substituting with NULL character, in the condition if there is another mark i.e. abovebase, belowbase, postbase precede the dot. We choose this condition to limit the deletion only to the case that 25CC is added in-between the two marks, and preserve the 25CC dot for the stand-alone mark (e.g. ീ ു ൂ ൃ) to prevent confusion.

```
feature calt{
  lookup delete_dot {
    sub [@abovebase @belowbase @postbase] uni25CC' by NULL; # globally
  } delete_dot;
} calt;
```

Testing words: ീ ു ൂ ൃ ൄ ൅

Dot should not be deleted from the stand-alone mark: ീ ു ൂ ൃ

C1.2 Spacing vowel sara-a

25CC dots must be deleted from the front side of sara-a ീ ു in any case, as these two glyphs have the same problem with the non-spacing mark (C1.1), when there is a belowbase or prebase preceding them and 25CC dot is automatically added.

However, there is a problem because there could also be several abovebase glyphs in-between the base and the belowbase e.g. ຈ ຈ ຈ ຈ. There can be many ways to fix this, but the most precise way (and most mundane way) is to add 1, 2, 3, ... abovebase arguments in the condition (3 should be enough).

```
lookup haang_ha_abv_blw {
    sub uni1A4C' @abovebase @belowbase by uni1A4C.alt;
    sub uni1A4C' @abovebase @abovebase @belowbase by uni1A4C.alt;
    sub uni1A4C' @abovebase @abovebase @abovebase @belowbase by uni1A4C.alt;
} haang_ha_abv_blw;
```

By the way, the substitution should not work with postbase ya ຢ [uni1A60 uni1A3F] because ya is a special character that needs a special rule (further discussion in CALT4, the postbase substitution). Therefore, the @postbase is omitted from the code.

```
lookup haang_ha_abv_blw {
    ignore sub uni1A4C' [uni1A3F.blw uni1A3F.blw.alt2];
    sub uni1A4C' @belowbase by uni1A4C.alt;
    sub uni1A4C' @abovebase @belowbase by uni1A4C.alt;
    sub uni1A4C' @abovebase @abovebase @belowbase by uni1A4C.alt;
    sub uni1A4C' @abovebase @abovebase @abovebase @belowbase by uni1A4C.alt;
} haang_ha_abv_blw;
```

We can use this format for the rest of the other glyphs in “haang” group.

```
feature calt {
lookup haang {

# no abovebase
sub uni1A2B' @belowbase by uni1A2B.alt;
sub uni1A2E' @belowbase by uni1A2E.alt;
sub uni1A4A' @belowbase by uni1A4A.alt;
sub uni1A4C' @belowbase by uni1A4C.alt;

# 1 abovebase
sub uni1A2B' @abovebase @belowbase by uni1A2B.alt;
sub uni1A2E' @abovebase @belowbase by uni1A2E.alt;
sub uni1A4A' @abovebase @belowbase by uni1A4A.alt;
sub uni1A4C' @abovebase @belowbase by uni1A4C.alt;

# 2 abovebase
sub uni1A2B' @abovebase @abovebase @belowbase by uni1A2B.alt;
sub uni1A2E' @abovebase @abovebase @belowbase by uni1A2E.alt;
sub uni1A4A' @abovebase @abovebase @belowbase by uni1A4A.alt;
sub uni1A4C' @abovebase @abovebase @belowbase by uni1A4C.alt;
```

```
# 3 abovebase
sub uni1A2B' @abovebase @abovebase @abovebase @belowbase by uni1A2B.alt;
sub uni1A2E' @abovebase @abovebase @abovebase @belowbase by uni1A2E.alt;
sub uni1A4A' @abovebase @abovebase @abovebase @belowbase by uni1A4A.alt;
sub uni1A4C' @abovebase @abovebase @abovebase @belowbase by uni1A4C.alt;
} haang;
} calt;
```

[illegible]

It should not work with: ဘဲ၊ ချို၊ စိမ်းစိမ်း၊ စိမ်းစိမ်း၊ စိမ်းစိမ်း၊ အေးအေး၊ အေးအေး၊ အေးအေး၊ အေးအေး

C2.2 Special case of Nya [uni1A2C]

Letter Nya နှ [uni1A2C] and the ligature nya-nya နှ [uni1A2C.1A2C] may need a special rule for the substitution. Its Haang does NOT need to be reduced in size if followed by postbase. This letter need a special rule that it should not be included in the class of @haang. It needs a separated group @Nya.

```
@nya = [unilA2C unilA2C.1A2C]

feature calt {
lookup nya_short_haang {

# nya-nya ligature 1A2C.1A2C, come first as the argument is longer
sub unilA2C.1A2C' @belowbase by unilA2C.1A2C.alt;
sub unilA2C.1A2C' @abovebase @belowbase by unilA2C.1A2C.alt;
sub unilA2C.1A2C' @abovebase @abovebase @belowbase by unilA2C.1A2C.alt;

sub unilA2C.1A2C' @abovebase @abovebase @abovebase @belowbase by
unilA2C.1A2C.alt;

# nya 1A2C
sub unilA2C' @belowbase by unilA2C.alt;
sub unilA2C' @abovebase @belowbase by unilA2C.alt;
sub unilA2C' @abovebase @abovebase @belowbase by unilA2C.alt;
sub unilA2C' @abovebase @abovebase @abovebase @belowbase by unilA2C.alt;



} nya_short_haang ;
} calt;
```

It should work with: ညှပ်နှိပ် ညှပ်နှိပ်

It should not work with: ဣ ညိ ညီ ဩ ညှိ ဩ ဣ ဣ ဣ ဣ ဣ

CALT 3 - Belowbase substitution (optional)

C3.1 Belowbase na-ma substitution

Belowbase form of na  and ma  [uni1A36.blw and uni1A3E.blw] may need to be shorten vertically to make a room for the second stack glyphs that come after or the Haang glyphs that come before. It can be done by CALT features as:

```
lookup na_ma_sub {
  # context before
  sub @haang uni1A36.blw' by uni1A36.blw.alt;
  sub @haang uni1A3E.blw' by uni1A3E.blw.alt;

  sub @haang @abovebase uni1A36.blw' by uni1A36.blw.alt;
  sub @haang @abovebase uni1A3E.blw' by uni1A3E.blw.alt;

  sub @haang @abovebase @abovebase uni1A36.blw' by uni1A36.blw.alt;
  sub @haang @abovebase @abovebase uni1A3E.blw' by uni1A3E.blw.alt;

  sub @haang @abovebase @abovebase @abovebase uni1A36.blw' by
  uni1A36.blw.alt;

  sub @haang @abovebase @abovebase @abovebase uni1A3E.blw' by
  uni1A3E.blw.alt;

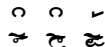
  # context after
  sub uni1A36.blw' [@belowbase @postbase] by uni1A36.blw.alt;
  sub uni1A3E.blw' [@belowbase @postbase] by uni1A3E.blw.alt;

  sub uni1A36.blw' @abovebase [@belowbase @postbase] by uni1A36.blw.alt;
  sub uni1A3E.blw' @abovebase [@belowbase @postbase] by uni1A3E.blw.alt;

  sub uni1A36.blw' @abovebase @abovebase [@belowbase @postbase] by
  uni1A36.blw.alt;

  sub uni1A3E.blw' @abovebase @abovebase [@belowbase @postbase] by
  uni1A3E.blw.alt;
} na_ma_sub ;
```

It should work with: 

It should not work with: 

C3.2 Second stack substitution: wa [uni1A45]

Sometimes, we may need to modify the dimension (reducing the size) of the belowbase glyphs in the second stack. The character wa (uni1A45) can act as a medial vowel and be placed in the second stack following another belowbase or postbase glyphs. We can follow the same template of the base substitution considering the addition of several abovebase glyphs in the contextual chain. We also need to consider the substitution after Haang group.


```
lookup secondstack_wa {
  sub [@belowbase @postbase @haang] uni1A45.blw' by uni1A45.blw.001;







  sub [@belowbase @postbase @haang] @abovebase uni1A45.blw' by
  uni1A45.blw.001;

  sub [@belowbase @postbase @haang] @abovebase @abovebase uni1A45.blw' by
  uni1A45.blw.001;

  sub [@belowbase @postbase @haang] @abovebase @abovebase @abovebase
  uni1A45.blw' by uni1A45.blw.001;
} secondstack_wa;
```

It should work with: 

It should not work with: 

* It is not canonical for the reduced size wa as the belowbase final consonant in the position of either the second stack ( rather than ) or the belowbase of Haang-group ( rather than ), but it's still possible to find it as it's more of a personal preference. The current shaping engine (2021) does not allow the mark positioning of the second stack wa if it's disrupted by the abovebase (e.g. unable to properly render ). However, wa as the medial in the second stack position () is the standard rule for Northern Thai spelling convention.

C3.3 Second stack substitution: sara-u [uni1A69, uni1A6A]

The same logic that the second stack glyphs should be reduced in size. Now, with the below vowel sara-u, sara-uu [uni1A69 uni1A6A].

```
lookup secondstack_u {
# Sara-u
  sub [@belowbase @postbase @haang] uni1A69' by uni1A69.alt;
  sub [@belowbase @postbase @haang] @abovebase uni1A69' by uni1A69.alt;

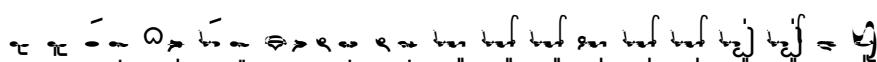
  sub [@belowbase @postbase @haang] @abovebase @abovebase uni1A69' by
  uni1A69.alt;

  sub [@belowbase @postbase @haang] @abovebase @abovebase @abovebase
  uni1A69' by uni1A69.alt;

# Sara-uu
  sub [@belowbase @postbase @haang] uni1A6A' by uni1A6A.alt;
  sub [@belowbase @postbase @haang] @abovebase uni1A6A' by uni1A6A.alt;

  sub [@belowbase @postbase @haang] @abovebase @abovebase uni1A6A' by
  uni1A6A.alt;

  sub [@belowbase @postbase @haang] @abovebase @abovebase @abovebase
  uni1A6A' by uni1A69.alt;
} secondstack_u;
```

It should work with: 

por pla

```
sub [@belowbase @postbase @haang] uni1A38.blw' by uni1A37.blw.alt;

sub [@belowbase @postbase @haang] @abovebase uni1A38.blw' by
uni1A37.blw.alt;

sub [@belowbase @postbase @haang] @abovebase @abovebase uni1A38.blw' by
uni1A37.blw.alt;

sub [@belowbase @postbase @haang] @abovebase @abovebase @abovebase
uni1A38.blw' by uni1A37.blw.alt;
```

} haang_ba_sub;

It should work with: ခ

It should not work with: ခ

Positioning (GPOS)

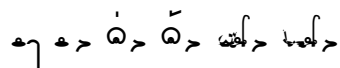
There are several ways to determine mark positioning. Several applications have automatic built-in kerning or mark positioning function that you don't need to code the GPOS table by yourself. This section will give you the guidance for the kerning or mark positioning rather than the full code in detail like in the previous GSUB section.

Tai Tham script can have the composition up to 3-4 abovebase glyphs. This is very complex and could be time consuming to kern or compose/decompose glyphs with such a degree of complex. In this case, mark positioning is useful for batch positioning, while GSUB ligature/composition is more practical for positioning a specific case (that doesn't fit with positioning in batch). However, a font with a careful design on the metrics might not need mark positioning.


POS 1 - Glyph class for mark positioning

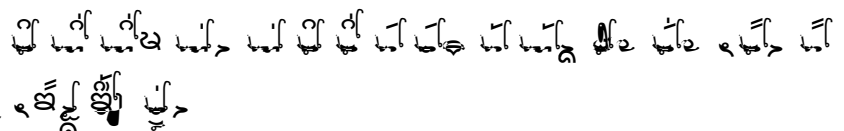
POS 2 - General mark-to-base positioning

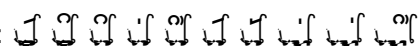
POS 3 - Sara-am positioning

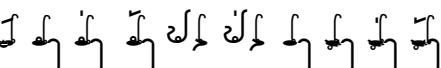
Rendering test : 

POS 4 - Mai Sam positioning

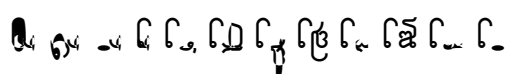
Rendering test, input maisat : 


Rendering test, input abv —> maisat : 

Rendering test, input mai-sat —> abv : 

Rendering test, input sara-am mai-sat —> abv : 

POS 5 - Kern for numerals

Numeral 2 : 

Numeral 7 : 

Limitations

Problems	Input order proposed by Unicode (2019)	Rendering by the current font (USE limitation)	Expectations
Second stack after abovebase	~ 𑖦 𑖧 𑖨 𑖩 𑖪	𑖪	𑖪 𑖩
Mai-sam as the syllable breaker	𑖪 𑖦 𑖧 𑖨 𑖩 𑖪 𑖫	𑖪𑖦𑖧𑖨𑖩𑖪	𑖪𑖪

These problems seem to be impossible to be fixed by the current Opentype rules (2021).