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

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	m	m	
Medial or Initial consonant cluster (either belowbase or postbase) (ccmp feature)	က ္လ သ ္ မ သ ္ ပ	න් (න්ව්) නී (නිව් ගී	Medial characters ൂ ു Hor-nam form ഗൃ ഗു ഗു ഗൂ ഗൂ ഗൂ ഗൂ And other semi-syllable cluster
Rawong	ო ((m	
Prebase vowels	က ြ ြ က ္လ ြ သ ္ ၾ ြ	ໄ (ຫ ໄ ຫຼ ໄ ລູ	ြ ေ
Belowbase vowels	က ္ ဝ က ္ည ဟ ္လ ွ ရ	දි ව ව ර	2865
Abovebase vowels	က ္ ိ ဟ ္ ဝ ိ သ ္ ၾ ေ ွ ိ	ర్మీ లీ కష్ట	ó 8 0 ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° ° °
Tone marks	က ္ ံ ံ ၿင ့္ ိံ	- ဧဦ ဧည့် ဧည့်	ပ်ပ်ပိတ်ပိ
Mai-Sam	သင္ဆႏွိွ္ပ္	ငည် ငည် ကို သို့ရ	Due to limitation with current shaping, Mai-sam can only be placed here, after finishing all abovebase sings.
Belowbase final consonant (sakot, liga feature)	៣ %្ឌ	င်ဗ္ဘ	
Postbase final consonant (sakot)	ત ૿ૺૄ လ	දී	
Spacing vowels (postbase vowels)	ພຸວໍຮ ພວໍສ	ස්ථි ස්ථි	
Sara-Am	ຕ ລ ໍ ຕ ່ ລ ໍ ຕ ັ ລ ໍ ຕ ູ ັ ລ ໍ ສ ຸ ຂ ລ ໍ	က်၁ က်၁ က်၁ က်၁ က်၁ ဆိ၁ ဆိ၁	
Raham / Karant	ຕ ૄ ບ ໍ້ ຕ ૄ ບ ໍ	ကူပံ ကူပံ	
Mai-Sam as word repeating sign	လ ု က ံ ်	လူကိ်	

Logical input order: Example for complex vowel forms

Vowel forms	Words	Input order
Sara i	ప్లో ఆ ర్యాం ర్య్ (fail to render)	သင္ လုိ ် ပ ဟင္ လုိ ် ဝ ဟင္ လုိ ် ္ ဝ (non-standard spelling)
Sara o	ြည့်	ပ [ု] ု ့ ့ ≊
Sara o (long)	ဏ္ထ်ရ သျှ်ရ	က _{လွ} ္ပ္ံရ သပ္ဃဉ္်ရ
Sara ɔ (short)	[ြင်း	က ြ ြ ့ ့ း
Sara x	ငော် ငော် ငော် ငော် ငော် ငေါ် ငေါ် ငေည် ငေည် ငေည် ငေည် ငေည် ငေည် ငေည် ငေည	က ေ ့ ိ က ္လ ေ ့ ိ သ ္ ၾ ေ
Sara wa	ငေရို့ အ ငေလို့] ငေရို့ န ငေလို့ န ငေလို့ န ငလို့ ကြွန်	ສ ຣ ຼ ິ ຕ ທ ຣ ຼ ິ ຳ ສ ກ ຸ ພ ຣ ຼ ິ ຳ ສ ນ ຸ ພ ຣ ຼ ິ ຳ ສ ປ ູ ຣ ຼ ິ ຳ ສ
Sara va	က္လွ တ္လွ ထို့ သျွ်ရ	ຕ ູ ္ ວ ຄ ທ ູ ຸ ວ ຄ ຕ ູ ຸ ວ ື ໝ ຸ ໙ ຸ ວ ີ ຄ
Sara am	స్త్రేర్ స్ట్రోం మ్ట్రోన్ల మ్ట్రోం ర్స్ జ్బ్	ນຸຂ່າ°່ ນຸຂ່າ°່ ນຸ ທ່າງ ຕ ນຸ ທ່າງ ຕ ວຸໄ° ສຸດ ວໍໄ°

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 ∞ uni1A31 + ∞ uni1A2C or ∞ uni1A2C + ∞ 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: pu pu pu m pu pu pu

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

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

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

Testing words: (జ్ఞం

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 ໝວນວຸລູບູຂວ.

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

Testing words: အသသဉ္ကပူနာ၁ ဝိရုဥ္က

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: || 4||
```

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

Sara-am has the input order for $\mathring{\mathbb{R}} = \mathcal{R} \circ \mathring{\circ}$ and $\mathring{\mathbb{R}} = \mathcal{R} \circ \circ \mathring{\circ}$. 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 $\mathring{\mathbb{R}} \mathring{\mathbb{R}}$ are required and must be placed before the whole Sara-am ligature.

L1.6 Sara-am ligature

Sara-am has the input order for $\mathring{\alpha} = Q \circ \mathring{\circ}$ and $\mathring{\dot{\alpha}} = Q \circ \mathring{\circ}$. Therefore it makes this ligature the most annoying one and causes headache the most. Also, don't forget to add ignore sub for $\mathring{\alpha}$ 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.

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

For stylistic aspect.

Testing words: ර ් ර් ර්

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: ឝ្នី ឝ្ន់ ញ្ចី ថ្នាំ

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: ຊື່ຊື່ ຶ່ວິນ
```

LIGA 3 - Ligatures with LookupFlag

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

Sara-u with belowbase ba uni1A37.blw ຈຼຸ ວຼຸ is not considered canonical. Normally it's written as ຈຼບ ວຼບ while sara-u with belowbase ya uni1A3F in a ligature form ල 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 unilA6A unilA60 [unilA37 unilA38] by unilA37.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: ဇာ တျှ တျှံ ဝာ့ နာ့ နာ့် နော့် နာ္ခံ ဇာ့်
```

L3.2 Letter with haag pa [uni1A5B]

This is for the stylistic aspect. But also, for the correct rendering, don't forget to add uni25CC in the argument as it can be inserted automatically.

```
lookup haang pa {
lookupflag UseMarkFilteringSet @belowbase;
# uni25CC removal
sub uni1A21 uni25CC uni1A5B by uni1A21.1A5B;
sub uni1A22 uni25CC uni1A5B by uni1A22.1A5B;
sub uni1A26 uni25CC uni1A5B by uni1A26.1A5B;
sub uni1A29 uni25CC uni1A5B by uni1A29.1A5B;
sub uni1A2A uni25CC uni1A5B by uni1A2A.1A5B;
sub uni1A2F uni25CC uni1A5B by uni1A2F.1A5B;
sub uni1A36 uni25CC uni1A5B by uni1A36.1A5B;
sub uni1A3E uni25CC uni1A5B by uni1A3E.1A5B;
sub uni1A47 uni25CC uni1A5B by uni1A47.1A5B;
sub uni1A4B uni25CC uni1A5B by uni1A4B.1A5B;
sub uni1A52 uni25CC uni1A5B by uni1A52.1A5B;
sub uni1A36 uni1A63 uni25CC uni1A5B by uni1A36.1A63.1A5B; # naa chrome ext, 25CC
removal
# normal
sub uni1A21 uni1A5B by uni1A21.1A5B;
sub uni1A22 uni1A5B by uni1A22.1A5B;
sub uni1A26 uni1A5B by uni1A26.1A5B;
sub uni1A29 uni1A5B by uni1A29.1A5B;
sub uni1A2A uni1A5B by uni1A2A.1A5B;
sub uni1A2F uni1A5B by uni1A2F.1A5B;
sub uni1A36 uni1A5B by uni1A36.1A5B;
sub uni1A3E uni1A5B by uni1A3E.1A5B;
sub uni1A47 uni1A5B by uni1A47.1A5B;
sub uni1A4B uni1A5B by uni1A4B.1A5B;
sub uni1A52 uni1A5B by uni1A52.1A5B;
sub uni1A36 uni1A63 uni1A5B by uni1A36.1A63.1A5B; # naa
} haang pa;
feature liga {lookup haang_pa;};
feature ccmp {lookup haang pa;};
```

L3.3 Belowbase form and postbase form of the letters CCMP vs LIGA

There are 2 ways to form a consonant cluster between base + belowbase glyphs: by the feature LIGA or the feature CCMP. To follow the logical input method, both CCMP and LIGA (or RLIG) are needed and they share several lookups. CCMP is require for the formation of the initial consonant cluster, while LIGA works for the formation of the final consonant belowbase/cluster. In brief, CCMP = initial consonant belowbase/cluster; LIGA = final consonant belowbase/cluster.

CCMP has a stronger joint, which is unbreakable (while it's the opposite for LIGA), so that CCMP is compatible with prebase glyph and can properly render cm ໄຫຼ . On the other hand, LIGA is not recognized

by the prebase glyph and will wrongly render the glyph cluster somehow like ຫຼຸເວ ຫຼຸໄ o 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 $\frac{\alpha}{2}$ (input $\alpha \circ \alpha \circ \alpha$) will be wrongly rendered somehow like $\frac{\alpha}{2} \circ \alpha$ 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 ບປ are the same as ୂ] ຼຸງ.
- 2) The belowbase form of uni1A40 $\[\[\] \]$ is in fact sign oy uni1A6D $\[\] \]$. Thus the belowbase substitution of uni1A40 is omitted.
- 3) The belowbase form of uni1A4B $\[mu]$ is in fact the sign o below uni1A6C $\[mu]$. 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 uni1A60 uni1A37 by uni1A37.blw;
sub uni1A60 uni1A38 by uni1A38.blw;
sub uni1A60 uni1A39 by uni1A39.blw;
sub uni1A60 uni1A3A by uni1A3A.blw;
sub uni1A60 uni1A3B by uni1A3B.blw;
sub uni1A60 uni1A3C by uni1A3C.blw;
sub uni1A60 uni1A3D by uni1A3D.blw;
sub uni1A60 uni1A3E by uni1A3E.blw;
# awak
sub uni1A60 uni1A3F by uni1A3F.blw;
#sub uni1A60 uni1A40 by uni1A40.blw;
sub uni1A60 uni1A41 by uni1A41.blw;
sub uni1A60 uni1A42 by uni1A42.blw;
sub uni1A60 uni1A43 by uni1A43.blw;
sub uni1A60 uni1A44 by uni1A44.blw;
sub uni1A60 uni1A45 by uni1A45.blw;
sub uni1A60 uni1A46 by uni1A46.blw;
sub uni1A60 uni1A47 by uni1A47.blw;
sub uni1A60 uni1A48 by uni1A48.blw;
sub uni1A60 uni1A49 by uni1A49.blw;
sub uni1A60 uni1A4A by uni1A4A.blw;
# sub uni1A60 uni1A4B by uni1A6C; # sing ow below
sub uni1A60 uni1A4C by uni1A4C.blw;
# sara loi (non standard)
sub uni1A60 uni1A4D by uni1A4D.blw;
sub uni1A60 uni1A4E by uni1A4E.blw;
sub uni1A60 uni1A4F by uni1A4F.blw;
sub uni1A60 uni1A50 by uni1A50.blw;
sub uni1A60 uni1A51 by uni1A51.blw;
sub uni1A60 uni1A53 by uni1A53.blw;
sub uni1A60 uni1A54 by uni1A54.blw;
} below_char;
# determine the features
feature liga { lookup below_char; } liga;
feature ccmp { lookup below_char; } ccmp;
It should not work with: ស្ស ធ្មូន 🤵
```

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.

Pre-Base form (pref)

By an unknown reason, the glyph Rawong uni1A55 (o dose not automatically work as the prebase glyph. It can be fixed by PREF feature.

```
feature pref {
lookup rawaong {
     sub uni1A55 by uni1A55;
} rawong; } pref;
```

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 standalone mark (e.g. $\centh{\circ}$ $\centh{\circ}$ $\centh{\circ}$) to prevent confusion.

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

Testing words: သျှိ သျွို သျွိုမ်း ဝိ် င်္ဂ

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 of 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.

```
feature calt{
        lookup delete_dot_aa {
            sub uni25CC' [uni1A63 uni1A64] by NULL;
      } delete_dot_aa;
} calt;
```

Testing words: သံ၁ သျှ၁ သျှံ၁ သျှံ၁ သျှံ၁ ဝါ ဝွှါ ဝွှ်ါဝွ်၁ င္ခ်ါ

C1.3 Consonant signs

```
lookup delete_dot_sign {
    sub @Letter uni25CC' [uni1A56 uni1A57 uni1A58 uni1A59 uni1A5A uni1A5B
    uni1A5C uni1A5D uni1A5E uni1A69 uni1A6A uni1A6C uni1A7F] by NULL;
} delete_dot_sign;
```

Testing words: လ၁ လ၁ လ၁ လ၁ လ၃ လ၃ ေလ် ေလ် ေလွ ေလွ ေလွ ေလွ ေလွ ေကွ ေတာ္လ လာ ေလွ် ေလွ် လွေ် ေလွ ေလွ် ေက္သံမ ကဉ္ က၃ က၃ ဝွ ဝွ

C1.4 Prebase glyphs

CALT 2 - Base substitution (shorten haang)

Base substitution must be the first before other substitutions, to make everything less complicated.

C2.1 Haang-group substitution

Some base glyphs have a descending line called "haang" i.e. \mathfrak{Q} \mathfrak{G} \mathfrak{G} \mathfrak{G} [uni1A2B uni1A2E uni1A4A uni1A4C]. This haang needs to be shorten to make a room for the below base glyph. For example, \mathfrak{G} \mathfrak{G} the "haang" of \mathfrak{G} should be shorten. This can be done by substituting the base glyph \mathfrak{G} with its alternate form with short-haang with the condition that there is a belowbase following it.

```
@haang = [uni1A2B uni1A2E uni1A4A uni1A4C uni1A2B.alt uni1A2E.alt uni1A4A.alt
uni1A4C.alt]
lookup haang_ha_blw{
      sub uni1A4C' @belowbase by uni1A4C.alt;
} haang_ha_blw;
```

```
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 j [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 unilA4A' @abovebase @abovebase @belowbase by unilA4A.alt;
sub uni1A4C' @abovebase @abovebase @belowbase by uni1A4C.alt;
```

C2.2 Special case of Nya [uni1A2C]

Letter Nya p [uni1A2C] and the ligature nya-nya p [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 = [uni1A2C uni1A2C.1A2C]
feature calt {
lookup nya short haang {
# nya-nya ligature 1A2C.1A2C, come first as the argument is longer
     sub uni1A2C.1A2C' @belowbase by uni1A2C.1A2C.alt;
     sub uni1A2C.1A2C' @abovebase @belowbase by uni1A2C.1A2C.alt;
     sub uni1A2C.1A2C' @abovebase @abovebase @belowbase by uni1A2C.1A2C.alt;
     sub uni1A2C.1A2C' @abovebase @abovebase @belowbase by
     uni1A2C.1A2C.alt;
# nya 1A2C
     sub uni1A2C' @belowbase by uni1A2C.alt;
     sub uni1A2C' @abovebase @belowbase by uni1A2C.alt;
     sub uni1A2C' @abovebase @abovebase @belowbase by uni1A2C.alt;
     sub uni1A2C' @abovebase @abovebase @belowbase by uni1A2C.alt;
} nya_short_haang ;
} calt;
It should work with: ည ည့် ည့် ည ဆွိ ဆွိ
lt should not work with: ည ည် ည် ညျှ ညျှ ည နည် နည် နည် နည္ခု နည္ခု ညှု ညှ
```

CALT 3 - Belowbase substitution (optional)

C3.1 Belowbase na-ma substitution

Belowbase form of na g and mag [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 Chaang Cabovebase Cabovebase Uni1A36.blw' by
    uni1A36.blw.alt;
    sub Chaang Cabovebase Cabovebase 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 not work with: တို့ တို့ တွ်
```

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' by uni1A45.blw.001;
} secondstack_wa;

It should work with: သွ သွ တွိ သျွ သျွ် သျွံ သျွံ သွံ နွ် တွိ သွံ ဝွ ေဆွ တွု တွ တွိ တွ
```

* It is not canonical for the reduced size wa as the belowbase final consonant in the position of either the second stack (600 rather than 600) or the belowbase of Haang-group (600 rather than 600), 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 600). However, wa as the medial in the second stack position (600) 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 unilA6A' by
      uni1A6A.alt;
      sub [@belowbase @postbase @haang] @abovebase @abovebase @abovebase
      uni1A6A' by uni1A69.alt;
} secondstack u;
```

CALT 4 - Postbase substitution

C4.1 Postbase Ya [uni1A3F]

Postbase glyph ya j [uni1A60 uni1A3F] can act as a medial vowel and needs to be elongated when following other belowbase or postbase glyphs (also haang-group). There are several conditions to be considered, i.e.

```
@haang = [uni1A2B uni1A2E uni1A4A uni1A4C uni1A2B.alt uni1A2E.alt uni1A4A.alt
uni1A4C.alt];
 feature calt{
 lookup postbase ya {
                              sub [@belowbase @postbase @haang] uni1A3F.blw' by uni1A3F.blw.alt2;
                              sub [@belowbase @postbase @haang] @abovebase uni1A3F.blw' by
                              uni1A3F.blw.alt2;
                              sub [@belowbase @postbase @haang] @abovebase @abovebase uni1A3F.blw' by
                             uni1A3F.blw.alt2;
                             sub [@belowbase @postbase] @haang] @abovebase @abovebase
                             uni1A3F.blw' by uni1A3F.blw.alt2;
 } postbase ya;
 } calt;
lt should work with: ဝှာရာ ဝွု၊ ဝွို၊ ဝွ်၊ ဝွ်၊ ဝွု၊ ဝျှ၊ ဝျှ၊ စ္ဆု၊ နွုံ၊ သျွန်၊ ဧည္ပို၊ ဧည္
 တ္ခ်ု ဇ႑ ဇ႑ နာ နာ့် အီြဋ္ဌာ တွာ တွံ
lt should not work with: ဝ၂ရာ ဝိ၂ ဝဴ၂ ဪ ဟုံ၂ ညာ ည်း) ဖွာ၁၂ ဖွာ၁၂ ဖွာ၁၂
```

C4.1 Postbase Ba [uni1A37, uni1A38]

The same manner as Ya.

```
lookup haang_ba_sub {
#long ba, in the same manner as long ya
#bo baimai

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

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

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

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

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 : ဝႆါ ဝံ၁ င်္ဂ၁ င်္ဂေ၁ ဆျီ၁ သျှီ၁

POS 4 - Mai Sam positionting

Rendering test, input mai-sat -> abv : သွိ သွိ သိ သိ သိ သိ သိ သွ် သွ် သွို သျှို သျှို

Rendering test, input sara-am mai-sat —> abv : නූ් රෝ ර්ෝ ර්ෝ ර්ෝ ර්ෝ රෝ රෝ රෝ ර්ෝ ර්

POS 5 - Kern for numerals

Numeral 2 : ၅ ၆၅ ၃၅ ဝ၂ ၆ ဩ ဖြ ဖြ ပြ ခြီ လြ ဝ

Limitations

Problems	Input order proposed by Unicode (2019)	Rendering by the current font (USE limitation)	Expectations
Second stack after abovebase	ဖ ္ၕိ်္ဝ	S _₽	cS ₅
Mai-sam as the syllable breaker	သဂ္ၾဴါိ္ဃ	ર્જૂ િ ુ	ર્યિટ્યું

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