Lao Encoding Dilemma and Moving onto Unicode

Phonpasit Phissamay[1], Nadir Durrani[2], Thonglor Duangsavanh[1]
STEA[1] , CRULP[2]
phonpasit@stea.gov.la[1] nadirdurrani@yahoo.com[2]

Abstract

This paper discusses the different popularly used ASCII based encodings in Lao. It also talks about the technique used to convert these encodings to Unicode.

1. Introduction

Like all other forms of information text is stored as a sequence of binary numbers. Character encoding is defined as assigning unique number to each language character to be processed by computer so that whenever a key is pressed from a keyboard or other input devices this particular code is generated internally in the computer.

ASCII (American Standard Code for Information Interchange) defined in 1968 by American National Standards Institute (ANSI) only contained English characters. This table was extended to 8 bits to include other Latin based languages spoken in Europe and South America but there was no entertainment to Lao and other Asian languages. Consequently to help themselves they exploit the ASCII encoding just replacing roman characters by Lao characters and using ASCII code table.

Till here it was a happy story but then there were problems to follow, due to lack of encoding standards. Arbitrary encoding may be defined for any application (e.g. 80 for letter 'ປ 81 for letter 'ອ). However this must be done according to some standards because if different vendors are defining encodings, their encoding may not agree with each other. This would mean that data generated through one application would mean something else when opened using application developed by another vendor who have used their own encoding. For example code 8081 would mean 'ປອ' when opened in application A and would mean 'ສ ຜ' in application B whose vendors have assigned codes 80 and 81 to 'ສ' and 'ຜ' respectively. Many coding conventions prevail and are being used to display Lao characters on DOS and Windows-based computers, some examples are IBM-81-1033, Alice, Hongkard etc. Each of these encoding do not comply with rest so the data saved with one appears garbage when viewed using other but all of these use one of the two following approaches with different encoding mechanism

- *Lower ASCII*

This was the first step as mentioned above it was simple technology that employed replacement of roman glyphs by Lao character glyph and using codes for English characters to write Lao text. The code started from 65 to onwards. The problem to this approach was if the user wanted mix Lao-English text he had to face problems switching fonts. Moreover there was not enough room for possible permutations and combinations of Lao consonants as they combined with tone marks and vowels.

- Upper *ASCII*

To solve the problem of mixed Lao-English text they used another mapping scheme not to use the numerals and English ASCII codes but to make use of the higher ASCII characters the 8th bit that was added to facilitate European and South American characters. In this fashion the user would be able to write English-Lao text with same font without having trouble to repeatedly change the font.

But there are still problems with both of these approaches, with each updated version of operating system there has to be an update and maintenance of font because position of new shape may be interrupted with some function in new operating system or application operating in the prohibited zone. And then Lao fonts do not work on systems on which Thai is enabled, and do not work at all on Japanese/Chinese/Korean versions of Windows.

[1] Science Technology and Environment Agency of Lao PDR
[2] Center for Research in Urdu Language Processing

2. Unicode

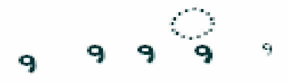
The advent of Unicode is on purpose to solve these encoding revolts that would have continued other wise not only for Lao encoding but for other languages that were using similar approaches and had no encoding standards. Word UNI means single so Unicode stands for single code to every character and no other character from any language could use that particular code. So no matter what platform, no matter what program, no matter what language you are using Unicode provides a two byte unique number for each character. Lao characters have been assigned space from 0xE000 to 0xF8FF.

3. Technique

The technique employed to this problem is fairly simple. A mapping table is defined that lists all the characters in all the three fonts like a character in higher ASCII, then character in lower ASCII and then character in Unicode like for example character ‘ກ’.

ກ ກກ

In case of more then one code for same character like in case of character ‘ ’ which has more then one code in higher ASCII all the possibilities have been listed that all these map on this particular character of Unicode :



In the above example first three characters belong to higher ASCII have different codes to represent the same character fourth character belongs to Unicode section and fifth belongs to lower ASCII section.
All the characters are listed in similar fashion. Program loads this table from mapping file then prompts the user to browse for input file and click open to let the program read the file.

3.1. Combinational vowels with tone marks

Unlike most of the glyphs that represent single character be it a consonant, vowel or a tone mark some of the glyphs like ‘ຳ້’ needs to be mapped on ‘◌ຳ’ and ‘◌້’ though we can find corresponding glyph in all the Lao fonts but the problem is that since this and similar other glyphs have no standard Unicode like other characters the font developer enjoys liberty of assigning code to these as per his desire. Consequently to make the program font independent we need to map such glyphs onto more then one character.

As the user selects a file program reads the file and decides if the text is in higher ASCII or Lower ASCII. File is read and stored as series of bytes in array which then compared with mapping table that was previously loaded from another file. So it traverses through the input list and compares code from mapping table and store corresponding Unicode character in output list. Like if the input file was lower ASCII first character was ‘ກ’ its code is 0x 0064 , the program looks for code 0x0064 in lower ASCII section and finds corresponding code in Unicode section which is 0xe81 and place it in output list. It traverses through entire input list to find corresponding matches. Other then Lao alphabets all the possible permutations like ‘ຳ້’ have been mapped one to one or one to many from both higher and lower ASCII to Unicode.

3.2. Latin characters

Latin characters, punctuation marks and other keys have same ASCII and Unicode just that ASCII is 8-bit and Unicode is 16-bit so all we have to do in case of Latin character is to put 00 for additional byte. So program could facilitate mixed English-Lao text as natural as Lao text. However there was problem in case of numerals in Lower ASCII font. The code for English numerals have been changed like for example code for ‘3’ is actually 0x0033 but this has been given 0x23 which is actually code for ‘#’ character in normal scenario therefore mapping for numerals have been explicitly defined and embedded within program for Alice (the Lower ASCII font).

3.3. Encoding Conversion

Although that Unicode encoding is here there is a serious hindrance for Lao people to adopt i.e. they are tuned to previous encoding system and the data they have saved in their previously being used encoding system. To solve problem number two we needed a conversion utility that could actually convert the Lao text written in ASCII format to that in Unicode format. Since dozen many encoding is being used it is hard to entertain all of these so we selected most popularly used encoding systems namely:

- Saysettha Lao
- Saysettha 2000
- IBM-STENO
- LAO ISO

- Alice_5

4. Conclusion

This document highlights the lack of encoding standard problems that prevail within Lao IT industry and explains how advent of Unicode has provided with potentially best solution but how could we preserve the data in old encoding format and convert it to new encoding scheme. An effort and one possible solution as a first step are discussed. Much work lies ahead for Lao script to create and mature standards required for ICTs (e.g. Keyboard, locale, etc.) and to develop meaningful applications and content around these standards which can eventually benefit Lao speakers.