The Khmer Script Tamed by the Lion (of TEX)

Yannis Haralambous
CERTAL (Centre d'Études et de Recherche sur le Traitement Automatique des Langues),
INALCO (Institut National des Langues et Civilizations Orientales), Paris, France.
Personal address: 187, rue Nationale, 59800 Lille, France
Internet: yannis@gat.citilille.fr

Abstract

This paper presents a Khmer typesetting system, based on TEX, METAFONT, and an ANSI-C filter. A 128-character of the 7-bit ASCII table for the Khmer script is proposed. Input of text is done phonically (using the spoken order consonant-subscript consonant-second subscript consonant-vowel-diacritic). The filter converts phonic description of consonantal clusters into a graphic TEXnical description of these. Thanks to TEX booleans, independent vowels can be automatically decomposed according to recent reforms of Khmer spelling. The last section presents a forthcoming implementation of Khmer into a 16-bit TEX output font, solving the kerning problem of consonantal clusters.

Introduction to Khmer Script

The Khmer script is used to write Khmer, the official language of the Cambodian Republic, and belongs to the Mon-Khmer group of Austroasiatic languages. It is a very old and beautiful script, and from the typesetter's point of view, one of the most challenging and exciting scripts in the world.

To understand the complications of Khmer typesetting, we will start with a quick overview of the Khmer writing system. Khmer is written from left to right; the Khmer alphabet has 32 *consonants*, the following:

ក ខ គ ឃ ង ច ឆ ជ ឈ ញ ដ ឋ ឌ ឍ ណ ត ថ ទ ធ
ន ប ផ ព ភ ម យ រ ល វ ស ហ ឡ

The character ◌ denotes the absence of a consonant. From the typesetter's point of view and with respect to collating order, it might as well be considered as a consonant. We will use a box □ to denote an arbitrary consonant.

These 33 "consonants" (except ឡ) can appear in the form of *subscript consonants:*

្ក ្ខ ្គ ្ឃ ្ង ្ច ្ឆ ្ជ ្ឈ ្ញ ្ដ ្ឋ ្ឌ ្ឍ
្ណ ្ត ្ថ ្ទ ្ធ ្ន ្ប ្ផ ្ព ្ភ ្ម ្យ ្រ ្ល ្វ
្ស ្ហ ្ឡ ្អ

A subscript consonant is pronounced after the "primary" consonant. Nevertheless, as the reader has certainly noticed, the subscript consonant ្□ is written on the left of the primary consonant.

It is also possible to have *two* subscript consonants carried by the same primary consonant. In that case, the second subscript consonant has to be ្ត. Examples: ទ្រ, ស្ត្រ.

A consonant, consonant + subscript or consonant + double subscript combination can carry a *vowel.* There are 28 vowels:

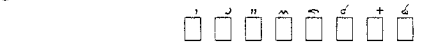
◌ា ◌ិ ◌ី ◌ឹ ◌ឺ ◌ឺ ◌ឹ ◌ុ ◌ូ ◌ួ ◌ើ ◌ឿ
◌ៀ ◌េ ◌ែ ◌ៃ ◌ោ ◌ៅ ◌ុំ ◌ាំ ◌ះ ◌ៈ

Although vowels are always pronounced *after* consonants, their graphical representation literally surrounds the consonant/subscript combination: they can appear above, beneath, on the right or on the left of consonants. Often a vowel's glyph has two or three non-connected parts.

When combining vowels with subscript consonants, the following graphical rules are followed:

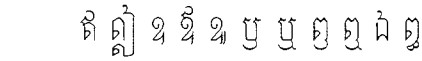
- if the subscript has a right protruding stem then the vowel ◌ា connects to the subscript and not to the consonant: ្ឋ + ◌ា = ្ឋា etc.
- if the consonant carries both a subscript ្□ and a vowel with left branch, then the latter is placed on the left of the former: ្□ + ◌ើ = ◌ើ្□ etc.
- if the consonant carries both a subscript consonant and a subscript vowel, then the latter is placed underneath the former: ្ក + ◌ុ = ◌ុ្ក, ្□ + ◌ួ = ◌ួ្□ etc.

Finally, a group of characters as described above can carry a *diacritical mark*. These are always placed above the character:



We will call the combination of consonant and eventual subscript consonant, second subscript consonant, vowel and diacritical mark, a *consonantal cluster*. Theoretically there can be 535,060 different consonantal clusters, but in practice less than 1% of them are really used. An analytic decomposition of A. Daniel's Khmer-French dictionary (Daniel, 1985) has provided no more than 2,821 different consonantal clusters out of 25,000 entries; colloquial Khmer may require even less clusters.

Besides consonantal clusters there are also 14 "stand-alone" characters in the Khmer alphabet:



These carry neither subscript consonants, nor vowels, nor accents. They cannot be found in subscript form. Orthographical reforms of Khmer have in some cases replaced them by "regular" consonantal clusters.

Inside a sentence, Khmer words are *not* separated by blank space. A blank space denotes the end of a sentence (or of part of a sentence: it plays the role of the period or of the semicolon in Latin script).

Hyphenation occurs between *syllables:* a syllable consists of one or two consonantal clusters with the sole restriction that the second cannot have a vowel. When a word is hyphenated, a hyphen is used. Sentences are "hyphenated" into words, but in that case, no hyphen is used. So from the typesetter's point of view, between two clusters hyphenation can be

1. forbidden (when the two clusters belong to the same syllable);
2. allowed and producing a hyphen (when the two clusters belong to the same word);
3. allowed without producing a hyphen (when the two clusters belong to different words in the same sentence).

This quick overview of the Khmer script has shown some of its particularities (see also Daniel (1985 and 1992), Tonkin (1991) and Nakanishi (1980)). To conclude, the author would like to underline the fact that the main difficulty in Khmer typesetting is the divergence between phonic and graphical representation of consonantal clusters (see Figure 1).

This paper is divided into five sections:

1. the definition and discussion of an 8-bit encoding table for information interchange and storage in the Khmer script. Consonantal clusters are encoded according to their phonic representation;
2. the presentation of three Khmer font families, designed in the METAFONT language. These fonts correspond to the three main styles of Khmer type and provide sufficient metaness to perform optical scaling, continuous interpolation from light to extra-bold weight and strong raster optimization;
3. the description of the process of deriving the graphical representation of consonantal clusters out of the phonic one (this process being implemented in an ANSI C preprocessor);
4. an overview of hyphenation and spelling reform rules and their realization in the preprocessor;
5. shortcomings of the Khmer typesetting system and plans for future developments.

The author would like to thank Prof. Alain Daniel (Institute of Oriental Languages and Civilizations, Paris) for his continuous support and encouragement and the Imprimerie Louis-Jean (Gap) in the person of Maurice Laugier, for having financed this project.

An 8-bit Encoding Table for the Khmer Script

Discussion. As mentioned in the introduction, Khmer language is written using consonantal clusters and stand-alone special characters. The collating order of consonantal clusters is given lexicographically out of the cluster components:

> Let $C_1 = c_1 s_1 s_1' \nu_1 d_1$ and $C_2 = c_2 s_2 s_2' \nu_2 d_2$ be two consonantal clusters, where $c_1, c_2 \in$ {consonants}, $s_1, s_2 \in \varnothing \cup$ {subscript consonants}, $s_1', s_2' = \varnothing$ or ◌្, $\nu_1, \nu_2 \in \varnothing \cup$ {vowels} and $d_1, d_2 \in \varnothing \cup$ {diacritics}. Then
>
> 1. $c_1 \succ c_2 \Rightarrow C_1 \succ C_2$;
> 2. if $c_1 = c_2$ then $s_1 \succ s_2 \Rightarrow C_1 \succ C_2$ (where $\varnothing$ precedes any other element);
> 3. if $c_1 = c_2$ and $s_1 = s_2$ then $s_1' \succ s_2' \Rightarrow C_1 \succ C_2$;
> 4. if $c_1 = c_2, s_1 = s_2$ and $s_1' = s_2'$ then $\nu_1 \succ \nu_2 \Rightarrow C_1 \succ C_2$;
> 5. if $c_1 = c_2, s_1 = s_2, s_1' = s_2'$ and $\nu_1 = \nu_2$ then $d_1 \succ d_2 \Rightarrow C_1 \succ C_2$.

The table of 128 codes for Khmer characters presented on the following page respects the collating order. Besides consonantal clusters and special characters, the following signs have been included in the 8-bit encoding:

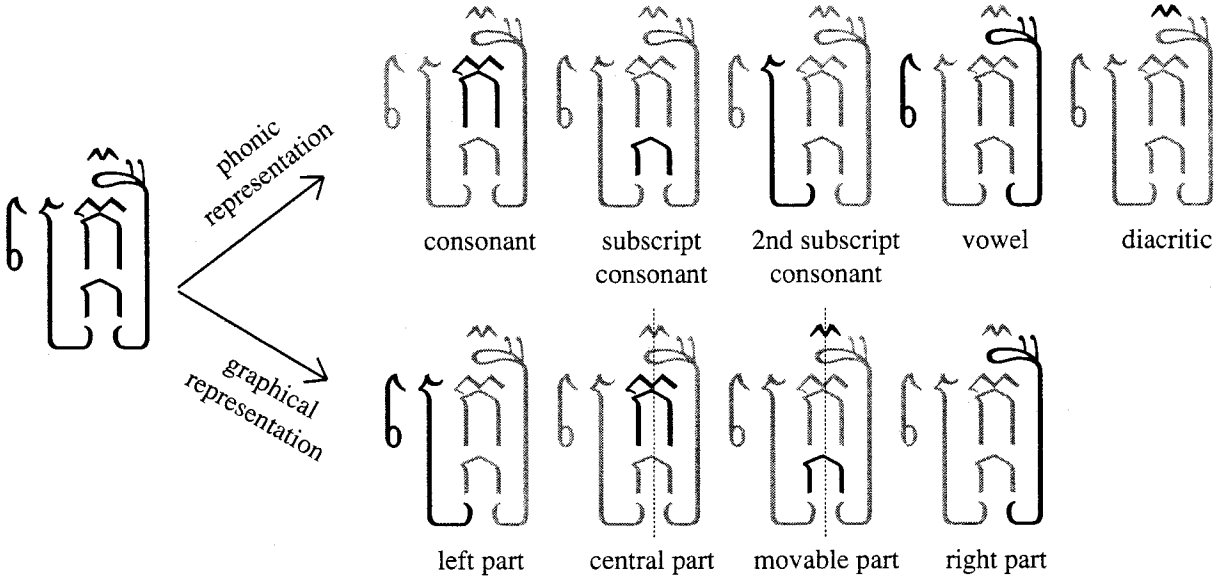


Figure 1: Decomposition of a Khmer consonantal cluster.

1. digits: ០, ១, ២, ៣, ៤, ៥, ៦, ៧, ៨, ៩;
2. punctuation marks other than the ones borrowed from Latin script: ។ (leikto), a variant form of the digit ២, indicating that the previous word is repeated (similar to Latin *bis*), ៕ (khan) and ៖ (bariyatosan), equivalent to a full stop, ៈ (camnocpikuh), a graphical variant of the Latin colon, and the French guillemets «, »;
3. the currency symbol ៛ (rial);
4. the invisible code WBK (word-break) to indicate the word limits inside a sentence.

Have *not* been included in the table:

- the archaic characters ឨ and ឩ which were abolished about a century ago;
- the punctuation marks ꕥ (cow's urine) and ꕤ (coq's eye), used in poetry, devination and classical texts;
- the variant forms ◌ឹ, ◌ឺ of ◌ឹ, ◌ឺ, used in *Dictionnaire Cambodgien* (1962).

These characters are nevertheless included in the TEX output fonts and can be accessed via macros.

The table. The table of codes 128–255 of the proposed 8-bit encoding for Khmer information interchange and storage follows. The 7-bit part of the table conforms to ISO 646 (standard 7-bit ASCII). Positions 0xCF and 0xDF are empty.

"8*	"9*	"A*	"B*	"C*	"D*	"E*	"F*
ក	ប	◌	◌	ឨ	០	◌	◌ាំ
ខ	ផ	◌	◌	ឩ	១	◌ា	◌ុំ
គ	ព	◌	◌	ឪ	២	◌ិ	◌ំ
ឃ	ភ	◌	◌	ឫ	៣	◌ី	◌ាំ
ង	ម	◌	◌	ឬ	៤	◌ឹ	◌ះ
ច	យ	◌	◌	ឭ	៥	◌ឺ	◌ៈ
ឆ	រ	◌	◌	ឮ	៦	◌ុ	«
ជ	ល	◌	◌	ឯ	៧	◌ូ	»
ឈ	វ	◌	◌	ឰ	៨	◌ួ	◌៉
ញ	ឝ	◌	◌	ឱ	៩	◌ើ	◌៊
ដ	ឞ	◌	◌	ឲ	។	◌ឿ	◌់
ឋ	ស	◌	◌	ឳ	ៗ	◌ៀ	◌៌
ឌ	ហ	◌	◌	឴	៘	◌េ	◌៍
ឍ	ឡ	◌	◌	឵	៙	◌ែ	◌៎
ណ	អ	◌	WBK			◌ោ	◌៏

Codes 0x80-0x9F and 0xC0 represent consonants; the corresponding subscript consonants are offset by 32 positions: they are represented by codes 0xA0 - 0xBE and 0xE0. The consonant 0x9F does not have a corresponding subscript consonant. The practice of subscripts having to be 32 positions apart from primary consonants is similar to the 32-position offset of uppercase and lowercase letters in ISO 646 (7-bit ASCII).

Codes 0xC0-0xCE represent special characters. Digits have been placed in positions 0xD0-0xD9, vowels in 0xE1-0xF5 and diacritics in 0xF8-0xFF. Finally, 0xFA is the currency symbol, 0xDB-0xDE are punctuation marks and 0xBF is the word-break code WBK.

Because of the 128-character limitation, vowels ◌ើះ, ◌ឿះ, ◌ុះ, ◌ើ◌ះ, ◌ា◌ះ, ◌ៅ◌ះ, ◌ា◌ៈ have not been included in the table. They have to be represented by the following code pairs:

◌ើះ	=	0xE2	0xF4	◌ឿះ	=	0xE4	0xF4
◌ុះ	=	0xE6	0xF4	◌ា◌ះ	=	0xE9	0xF4
◌ៅ◌ះ	=	0xEC	0xF4	◌ៅ◌ះ	=	0xED	0xF4
◌ា◌ៈ	=	0xEF	0xF4				

Requirements for Khmer script software. As in the case of Arabic and Hindi, software displaying Khmer text has to provide context-analytic algorithms. Following is an exhaustive list of the necessary context-dependent transformations:

1. when code 0xBA follows a code in the range 0x80-0x9E, 0xC0 then their glyphs must be permuted, e.g., ក + ◌ុ → ◌ុក.
2. when code 0xBA follows a pair of characters $\alpha\beta$, with $\alpha \in$ {0x80-0x9E, 0xC0}, $\beta \in$ {0xA0-0xBE, 0xE0} then the glyph of 0xBA must appear on the left of the glyphs of α, β, e.g., ក + ◌្ + ◌ុ → ◌ុក្.
3. when codes 0xE9-0xEC and 0xEF-0xF0 follow a combination of character codes $\alpha, \alpha\beta, \alpha\beta\gamma$ where α and β are as in the previous item and γ =0xBA, then the glyph ◌ must appear on the left of the latter combinations. Example: ក + ◌្ + ◌ុ + ◌ា → ◌ាក្ក.
4. when codes 0xED and 0xEE follow a combination $\alpha, \alpha\beta, \alpha\beta\gamma$ of codes as in the previous item, then their glyphs must appear on the left of these combinations;
5. when code 0x89 (ញ) is followed by a code in the range 0xA0-0xBE, 0xE0 then the variant glyph ញ must be used. Example: ញ + ◌ូ → ញ◌ូ.

When the second code is 0xA9 then a variant glyph must be used for it as well: ញ + ◌្ → ញ◌្.

These contextual transformations have been implemented by the author into a modified version of the Macintosh freeware text editor Tex-Edit by Tim Bender, included in the package. In Figure 2 the reader can see the effect of striking successively keys <ក>, <◌្> (<subscript modifier> followed by <ឃ>), <◌ុ> (<subscript modifier> followed by <ះ>), <◌ា>, to finally obtain the consonantal cluster ◌ាក្ក.

METAFONTing Khmer

Font styles. There are three styles used in Khmer typesetting: standing (aksar ch-hor), oblique (aksar chrieng) and round (ak-sar mul). The latter is virtually identical with inscriptions of the 12th and 13th centuries at Angkor Wat and is reserved for religious texts, chapter headings, newspaper headlines, inscriptions and on other occasions where it is wished to make a contrast with the oblique script, to add a touch of formality, or to provide variation of emphasis (see Tonkin (1991)).

The author has designed three METAFONT font families, corresponding to these styles; samples of these fonts in 14.4 point size can be seen on Figure 3; Figure 4 shows a sample headline using the round font.

In Figure 5, the Khmer letter ក has been reproduced 256 times, with different values of two parameters: the widths of "fat" and "thin" strokes. The central vertical symmetry axis represents "Égyptienne"-like characters, where the parameters have the same value. This classification can of course be refined and allows an arbitrarily precise choice of the font gray density.

Obtaining a Graphical Representation out of a Phonic One

Above we have given a quick overview of the (minimal) contextual analysis involved in displaying Khmer script on screen. The situation is much more complicated in the case of high quality typesetting.

TEX is the ideal tool for typesetting in Oriental scripts like Khmer, because of the inherent fundamental concept of *boxes* (see *The TEXbook* (Knuth, 1989) and Kopka (1991 and 1992)). As in mathematical formulas, elements of a consonantal cluster are moved to aesthetically correct positions and then grouped into a single and indivisible "box" which TEX treats as a single entity.

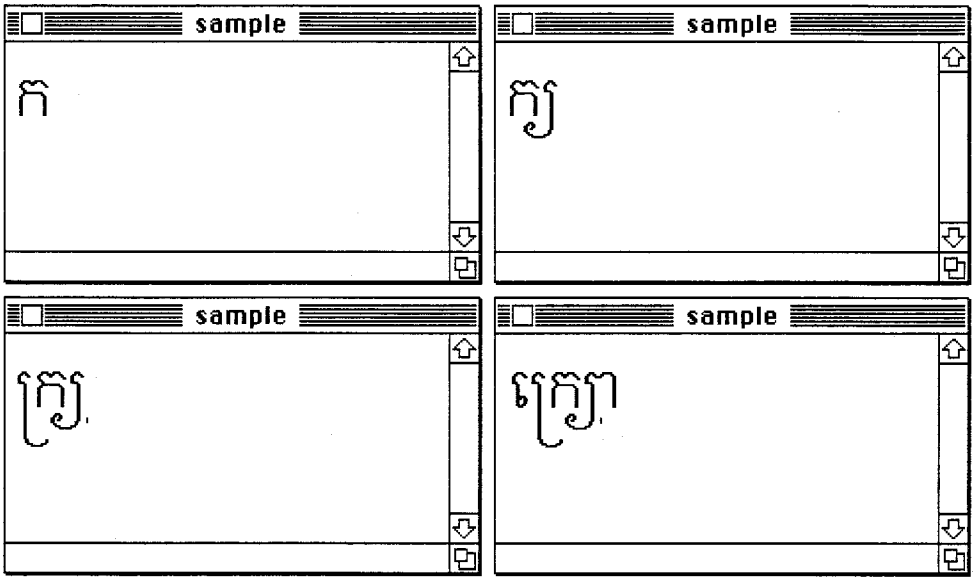


Figure 2: A text editor with Khmer contextual properties.

Standing characters

នៅឆ្នាំ ១៩៦២ ប្រទេសកម្ពុជាច្បាស់ចេញមក នៅសល់តែស្បែកនឹងឆ្អឹង អំពីការប្រយុទ្ធប៉ុយួរអង្វែង
គឺជាដំយោរ យៅមួយ ប្រធាំងនឹងការជិះជាន់របស់ឃ្មុន គឺជាសៀម ។ ការឈឺចាប់នៃប្រជាពលរដ្ឋយើង-
នៅពេលនេះ គឺហួសពីការគិតទៅហើយ ប៉ុន្តែពួកបស្ចិមប្រទេស គេនៅតែពុំដឹងឮ ចំពោះការឈឺចាប់-
នេះទេ ។ ធំណែកខ្មែរវិញ យើងមិន មានភ្លុចសោះឡើយ ។

Oblique characters

*នៅឆ្នាំ ១៩៦២ ប្រទេសកម្ពុជាច្បាស់ចេញមក នៅសល់តែស្បែកនឹងឆ្អឹង អំពីការប្រយុទ្ធប៉ុយួរអង្វែង គឺជាដំ-
យោរ យៅមួយ ប្រធាំងនឹងការជិះជាន់របស់ឃ្មុន គឺជាសៀម ។ ការឈឺចាប់នៃប្រជាពលរដ្ឋយើងនៅ-
ពេលនេះ គឺហួសពីការគិតទៅហើយ ប៉ុន្តែពួកបស្ចិមប្រទេស គេនៅតែពុំដឹងឮ ចំពោះការឈឺចាប់នេះទេ
។ ធំណែកខ្មែរវិញ យើងមិន មានភ្លុចសោះឡើយ ។*

Round characters

**នៅឆ្នាំ ១៩៦២ ប្រទេសកម្ពុជាច្បាស់ចេញមក នៅសល់តែស្បែកនឹងឆ្អឹង អំពីការប្រយុទ្ធប៉ុយួរអ-
ង្វែង នឹងដំយោរ យៅមួយ ប្រធាំងនឹងការជិះជាន់របស់ឃ្មុន នឹងសៀម ។ ការឈឺចាប់នៃប្រ-
ជាពលរដ្ឋយើងនៅពេលនេះ គឺហួសពីការគិតទៅហើយ ប៉ុន្តែពួកបស្ចិមប្រទេស គេនៅតែពុំ-
ដឹងឮ ចំពោះការឈឺចាប់នេះទេ ។ ធំណែកខ្មែរវិញ យើងមិន មានភ្លុចសោះឡើយ ។**

Figure 3: Samples of Khmer fonts.

Figure 4: Headline in round style.

In this section we will see how the graphical representation of a cluster is constructed, using both the preprocessor and TEX.[1]

Graphical classification of Khmer cluster components. As already mentioned, there is a strong divergence between the phonic and graphical representation of a consonantal cluster: for example, if c =ឈ, s_1 =◌្ឈ, s_2 =◌្ញ, v =◌ើ, then for the same cluster ញ្ញ្ញើ, the former representation is <c><s_1><s_2><v> and the latter <v (left branch)><s_2><c><s_1><v (right branch)>.

A thorough study of Khmer script and traditional typography has resulted in the following classification of graphical components of a consonantal cluster:

1. **the "left part".** Four elements which are placed on the left of a consonant: ◌ុ, ◌ូ, ◌ួ, ◌ើ.
2. **the "central part".** All consonants: ក, ខ ... ឣ. Also consonant + vowel ∈ {◌ា, ◌ិ, ◌ី} combinations, whenever the vowel is attached to the consonant and not to a subscript: កា, កិ, កី etc. but *not* ក្ញ.

   The difference between "left" and "central" part is that only the latter is taken into account when determining the symmetry axis of the cluster.
3. **the "movable" part.** Subscripts and superscripts which are moved horizontally so that their symmetry axis coincides with the axis of the central part: ◌្ក ... ◌្ឣ, and ◌ំ, ◌ះ, ◌ៈ, ◌ៃ, ◌ោ, ◌ៅ, ◌ុំ, ◌ូំ, ◌ួំ, ◌ើំ, ◌ើះ.
4. **the "right" part.** Elements placed on the right of the central part, and not involved in the determination of the cluster symmetry axis. In this category we have certain subscript characters: ◌្ឋ ... ◌្ឡ, as well as selected subscript and superscript vowels and diacritical marks: ◌ុ, ◌ឹ, ◌ឺ, ◌ះ, ◌ៈ, ◌ៃ, ◌ោ, ◌ៅ, ◌ំ.

The effective graphical construction of a consonantal cluster by TEX is done in the following way: the preprocessor's output replaces the phonic representation of a cluster (in the encoding described under Discussion) by a TEX macro `\ccl` with 5 arguments: the first is a 9-digit number representing the phonic representation of the cluster (and with the property that if N, N' are numbers representing clusters C, C' then $C \succ C' \Leftrightarrow N > N'$, where $\succ$ is the collating order of clusters and $>$ the usual ordering of integers); the remaining four correspond to the four parts of the graphical decomposition of a cluster as described above. For example,

```
\ccl{050311501}{e/r}{gA}{/k}{'}
```

indicates a left part `e/r` (ឬ◌), a central part `gA` (ឣ), a movable part `/K` (◌្ក) and a right part `'` (◌ំ). This example illustrates the important fact that the symmetry axis of the central part is not necessarily the middle axis of the box containing the central part:

ឬ្ឣំ and not ឬ្ឣំ

The difference is of more than just aesthetic nature: in some cases the vertical alignment of elements of a cluster is necessary to determine the cluster itself. Take for example characters 0x89 (ញ) and 0x96 (ទ). When the latter is followed by a vowel ◌ើ it becomes ញ, which is indistinguishable from the upper part of the former: it is the lower part

[1] Theoretically it would be possible for TEX to graphically represent a cluster out of its phonic representation without the assistance of a preprocessor. However, this process would be too slow and memory-consuming for real-life typesetting.

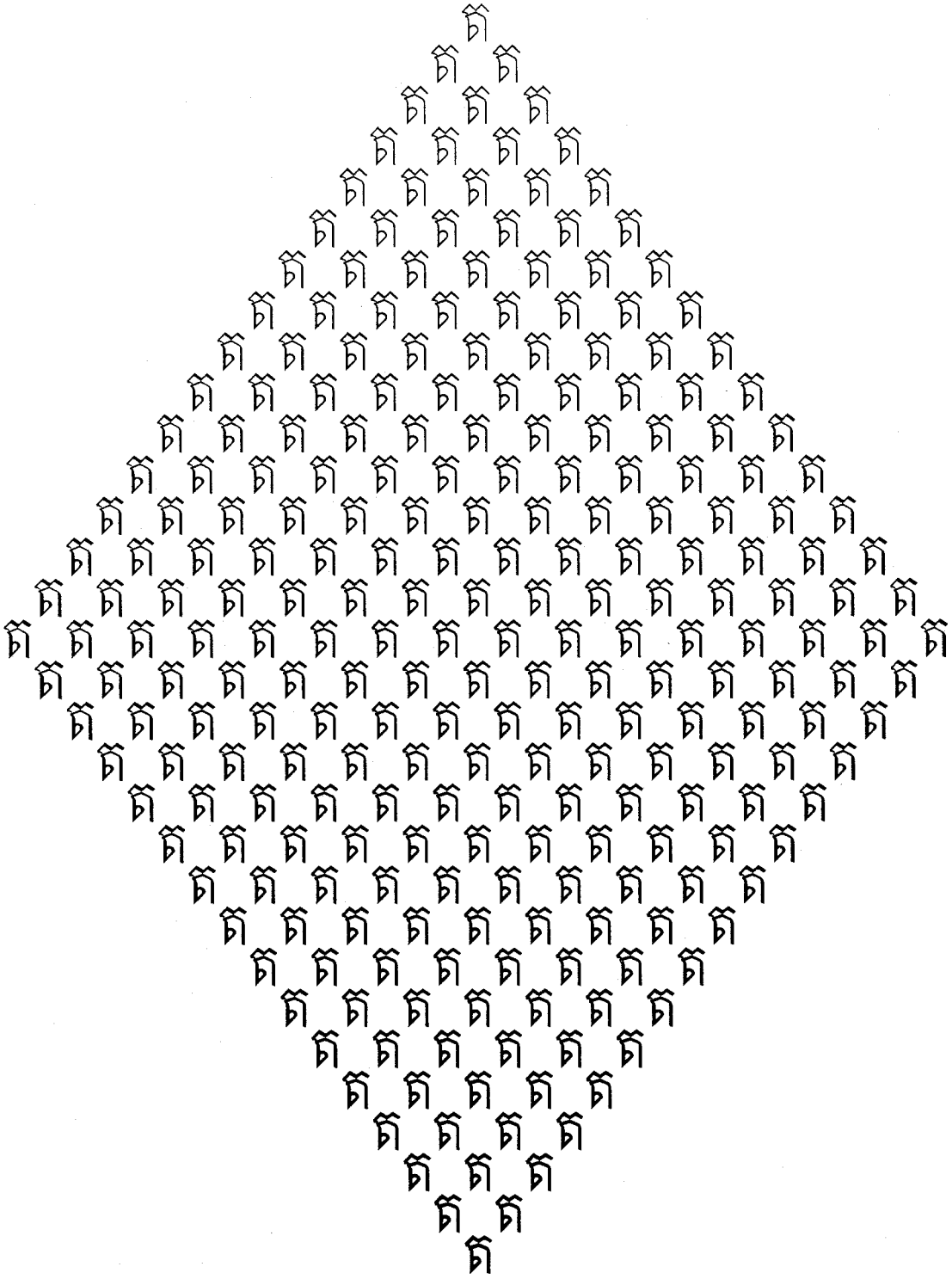


Figure 5: Test table for gray density fine-tuning of Khmer font.

that allows differentiation. But when both happen to carry the same subscript consonant then this lower part vanishes. The difference will be found in the alignment of the subscript consonant: in the case of one would have for example , while in the case of it would be .

From these considerations we conclude that the symmetry axis location is a vital piece of information for every character; it depends on the shape of the individual character and *cannot* be given by a general font-independent rule.

In TEX one has several global parameters for a given font, but only 4 for every individual character of the font: width, height, depth, italic correction. The author has used the parameter "italic correction" as a carrier of the information on the symmetry axis location.

The construction mechanism is very simple: TEX typesets first the left part and the central part of the cluster; then it moves to the left, by an amount equal to the italic correction of the central part and typesets the movable part; finally it moves back to the right edge of the central part and typesets the right part of the cluster.

To simplify this mechanism, all movable elements are of zero width. The reader can see an example in Figure 6, where TEX boxes are displayed in gray and the symmetry axis of the central part by means of a dotted line.

Special cases and exceptions. The mechanism of cluster construction described above fails in certain special cases. These are handled by using variant forms of graphical elements. Following is a quick description of these cases.

1. often two or three subscripts or superscripts are found in the same cluster. In these cases the following rules apply:
   (a) in the case of two subscript consonants, the second being necessarily , a deeper form of the latter is used: + = ;
   (b) in the case of a subscript consonant and a subscript vowel, the vowel is placed under the subscript consonant: + = . This rule applies also for the subscript consonant ; + = ;
   (c) in the case of two subscript consonants and a subscript vowel, the consonants are placed as in (a) and the vowel is placed on the right of ; + + = ;
   (d) in some cases we have both a superscript vowel and a diacritical mark. The following combinations are known: , , , , , , ;
2. to prevent confusion between the letter followed by vowel , and the letter , the former combination of consonant and vowel is written . A variant of this letter is used in the presence of a subscript: + = ;
3. when a cluster with contains vowel or , then the width of the primary consonant determines the depth of the vowel: + + = , but + + = ;
4. the letter is not supposed to carry a subscript consonant; in some rare cases, it carries subscript : .

Collating order. As mentionned in the previous section, the TEX command `\ccl`, obtained by the preprocessor, describes a cluster by means of five arguments. The last four arguments describe the cluster graphically: they correspond to the four parts of the graphical decomposition of a cluster, according to the subsection on Graphical classification of Khmer cluster components. The first argument corresponds to the phonic decomposition of the cluster; it is a 9-digit number $N = c_1c_2s_1s_2s_3v_1v_2d_1d_2$ where

1. c_1c_2 determines the primary consonant of the cluster: c_1c_2 goes from 01 = , to 33 = ;
2. s_1s_2 determines the (first) subscript consonant: s_1s_2 = 00 if there is no subscript consonant, otherwise s_1s_2 goes from 01 = , to 32 = ;
3. s_3 = 0 if there is no second subscript consonant, 1 if there is a second subscript ;
4. v_1v_2 determines the vowel: v_1v_2 = 00 if there is no vowel, otherwise v_1v_2 goes from 01 = , to 28 = ;
5. d_1d_2 determines the diacritical mark: d_1d_2 = 00 if there is no diacritic, otherwise d_1d_2 goes from 01 = , to 08 = .

A complete list of characters, alphabetically ordered, is given in the Introduction. From the rules of collating order, it follows that for clusters C, C' and their corresponding 9-digit numbers N, N', we have

$$C \succ C' \Leftrightarrow N > N'$$

where $\succ$ is the collating order of clusters. The numbers N, N' can be easily ordered since the collating order of clusters corresponds to their order as integers. This fact allows straightforward searching, sorting, indexing and other collating order involving operations.

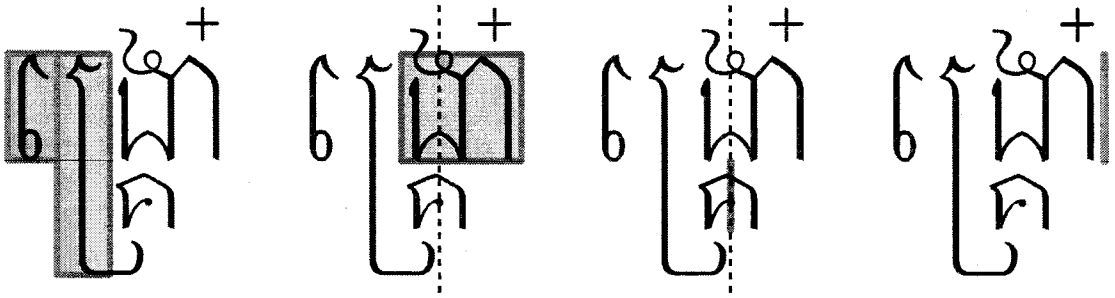


Figure 6: Construction of a Khmer consonantal cluster by TEX.

Hyphenation and Other Preprocessor Features

Hyphenation. Hyphenation of Khmer obeys a very simple rule: words are hyphenated between syllables. Unfortunately this rule can hardly be implemented by a computer since there is no algorithmic way of detecting syllables: a syllable can consist of one *or two* consonantal clusters.

With the help of Prof. Alain Daniel an empirical hyphenation mechanism has been developed, out of several general rules and observations. Following is a first set of rules — there will be further refinement after thorough testing on bigger amounts of Khmer text.

> Let C, C' be consonantal clusters. Hyphenation C-C' is possible whenever:
>
> 1. C' contains a vowel;
> 2. C contains a vowel among ◌ិ, ◌ី, ◌ុ, ◌ឹ, ◌ឺ, ◌ួ, ◌ើ, ◌ឿ, ◌ៀ, ◌េ, ◌ែ, ◌ៃ, ◌ោ, ◌ៅ, ◌ំ, ◌ះ, or one of the diacritical marks ◌៉, ◌៊;
>
> Hyphenation is always possible before or after special characters.

TEX provides an internal hyphenation mechanism based on *hyphenation patterns*. Unfortunately this mechanism cannot be used in the case of Khmer consonantal clusters, since these are enclosed in boxes and hence cannot be considered as characters by TEX. For this reason, the hyphenation algorithm is performed by the preprocessor; whenever one of the two rules above is satisfied, the TEX macro `\-` is included in the output. This command expands as

```
\def\-{\discretionary{-}{}{}}
```

so that a hyphen is obtained whenever a word is hyphenated. There is no algorithm yet for automatic decomposition of sentences into words: the user is asked to include `WBK` (word-break) codes between words inside a sentence. These codes are converted into `\wbk` commands by the preprocessor; `\wbk` expands into

```
\def\wbk{\discretionary{}{}{}}
```

that is: a potential hyphenation point, *without* hyphen.

Decomposition of special characters and spelling reforms. The special characters (codes `0xC1-0xCE`) are mostly historical residues and loans from other languages (Pali and Sanskrit). There have been many attempts by the Cambodian Ministry of Education to restrain their number, by eventually replacing some of them with regular consonantal clusters.

This replacement can vary from word to word. Prof. Alain Daniel has established a list of reformed words and their replacements. This list is known by the preprocessor, which will output every special character as a TEX macro with a numeric argument, indicating the potential replacement by some other special character or by a consonantal cluster. For example, according to the surrounding word, ឲ is output as `\ao0`, `\ao1`, `\ao2`, `\ao3` or `\ao4`. If a certain boolean variable `\ifreformed` is false then all five macros will always expand into ឲ. On the other hand, if the boolean is true, then the first macro will expand into ឱ, the second into ឣ, the third into ឧ, the fourth into ឪ and the fifth into ឣៅ.

Following is a first list of reformed words, known by the preprocessor. The special characters and their decompositions are set in bolder type.

រំពឹស → រ**ំអ**ិស	រពឹស → រ**អ**ិស
ពិសុការ → ឲ**ស**ុការ	ឦស → ពិស
ឦសធរ → ពិសធរ	ឦសាធ → ពិសាធ
ឦស្សរ → ពិស្សរ	ឦស្សរៈ → ពិស្សរៈ
ឲក → **អ**ុក	ឲស → **អ**ុស
ក្រឲ → ក្រ**អ**ៅ	ក្រាំងក្រឲ → ក្រាំងក្រ**អ**ៅ
ឲកា → ឱ**ក**ា	ឲដ្ឋ → **អ**ូដ្ឋ
ឲធ → **អ**ូធ	ឲម! → ឱ**ម**!

ឧរុ → ឱរុ	ឧរូ → ឱរូ
ឬ → រឹ	សំឬទ្ធ → សំរឹទ្ធ
រឦក → រលឹក	រឦក → រលឹក
ឮ → លឺ	ឮយ៉ង → លឺយ៉ង
ឮជា → លឺជា	ឮលាន់ → លឺលាន់
ឮសាយ → លឺសាយ	ឮរវៃណ → អៃរវៃណ
ឮ! → អៃ!	ឮក- → អៃក-
ឮក្ស- → អៃក្ស-	ឮរវៃត → អៃរវៃត
ឮស្សរ → អៃស្សរ	ឮស្សរ្យ → អៃស្សរ្យ
ប្រឱង → ប្រអោង	រឱក → រអោក
សំឱន → សំអោន	សំឱក → សំអោក
ឱរែ → ឱរែ	ឱ! → អៅ
ឱងឡោង → អោងឡោង	ឱង្គា → អង្គា
ឱបួសន៍ → ឱបួសន៍	ឱត → អោត
ឱឡាន → ឱឡាន	ឱបច្ចាតិក → ឱបច្ចាតិក
ឱរែ → ឱរែ	ឱសហ៍ → ឱសហ៍
ឱ → អៅ!	ឱទក → ឱទក

Shortcomings and Plans for Further Development

The system presented in this paper allows high quality Khmer typesetting. It is the first Khmer typesetting system resolving problems such as text input in phonic order, positioning of subscripts and superscripts, optical scaling, hyphenation and replacement of special characters.

Nevertheless the graphical cluster-construction algorithm as described in this paper has certain flaws; a few examples:

- if a consonant with subscript consonant carries the ◌ុ vowel, then the latter should be justified at the right edge of the *subscript*, which is not necessarily aligned with the right edge of the consonant. For example, in the (hypothetical) cluster ន្ត្ុ, the ◌ុ is badly positioned;
- take a narrow letter (like រ, ឫ) which carries a large subscript (like ◌្ក or ◌្ល) and suppose you are at the line boundary (either left or right); then contrary to the normal use of subscripts, it is the subscript which should serve for line justification, and not the consonant.

These problems cannot be solved using the current mechanism (in which TEX considers that all subscripts and superscripts are of zero width). It could be possible to use subscripts with non-zero width, but (a) this would slow the process down, and (b) it wouldn't solve the problem of the line boundary, since we are asking for contradicting properties: inside a sentence subscripts should not interfere in determining the distance between clusters, while at the line's boundary they should.[2] Furthermore, one could imagine a sentence ending with ន្ត្ុ and the next sentence starting with រ្ក. The blank space in-between is hardly sufficient to prevent clusters from overlapping... visually the beginning of the sentence is lost.

Corrections to these problems can be done manually (after all these problems occur very rarely). But a much more natural and global solution would be to treat consonantal clusters as individual codes in a 16-bit encoding scheme. As mentioned in the introduction, only 2,821 clusters (out of 535,000 theoretical possibilities) have been detected in the fairly complete dictionary of Prof. Alain Daniel, so a 16-bit table would be more than sufficient to cover them.

Text input could still be done using the 8-bit encoding of Section 1; the preprocessor would then convert the 8-bit description of consonantal clusters into their codes in the 16-bit table (or by their explicit construction, if for any reason they are not included in the table). This approach is similar to Kanji construction out of Kana characters in Japanese, or to Hangoul construction out of elementary strokes in Korean.

An extended version of TEX —which according to D.E. Knuth's directives should not be called TEX anymore— would then perform real kerning and hyphenation, since in this case consonantal clusters would be treated by TEX as *letters*. Work in the direction of an extended TEX is currently being done by Philip Taylor[3] and his team (NTS project), and by John Plaice[4] (Ω project)[5]

[2] Unfortunately, in TEX there is no such thing as an `\everyline` command.

[3] Royal Holloway College (UK)

[4] Université Laval (Canada)

[5] It should be mentioned that the Japanese TEX Users Group, and the ASCII Corporation, Kanawa-sakishi Kanagawa, Japan, have developed and released 16-bit versions of TEX. Unfortunately, these are not "real" 16-bit TEXs (and hence inefficient for 16-bit Khmer), because they allow only 256 different character widths (it happens that Japanese Kanji, just like Chinese characters, have all the same width) and 256^2 kerning pairs or ligatures. True 16-bit TEX should allow $256^2 = 65,536$ character widths, and $65,536^2 = 4,294,967,296$ kerning pairs or ligatures. Also 16-bit hyphenation patterns should be possible. Besides Khmer, such an extended TEX version would be extremely useful for ligatured Arabic Naskhi, Urdu Nastaliq, Hebrew with cantillation marks, and other scripts with "advanced typographical requirements".

By using virtual property lists, no additional bitmap files would be added. The 16-bit font would be made by virtually assembling glyphs taken from the already available 8-bit font.

Availability

The METAFONT, TEX and C sources of all software presented in this paper belong to the **public domain**. They constitute a proposal for a Khmer *TEX Language Package*, submitted to the Technical Working Group on Multiple Language Coordination of the TEX Users Group and will be released after ratification. The α version of the package is currently being tested in Cambodia, and can be obtained by the author (yannis@gat.citilille.fr). This work will be presented either by Alain Daniel or by the author at the *First Conference on Standardization of Khmer Information Interchange*, organized by UNESCO, July 20–23 in Phnom Penh, Cambodia.

Bibliography

Daniel, Alain. *Dictionnaire pratique cambodgien-français.* Institut de l'Asie du Sud-Est, Paris, 1985.

Daniel, Alain. *Lire et écrire le cambodgien.* Institut de l'Asie du Sud-Est, Paris, 1992.

វចនានុក្រមខ្មែរ, ការផ្សាយរបស់ពុទ្ធសាសនបណ្ឌិតយ, ព. ស. ២៥០៥ [*Dictionnaire Cambodgien.* Éditions de l'Institut Bouddhique, Phnom-Penh, 1962.]

Knuth, Donald E. *The TEXbook.* Computers and Typesetting, Volume A. Addison-Wesley, Reading, 1989.

Knuth, Donald E. *The* METAFONT*book.* Computers and Typesetting, Volume C. Addison-Wesley, Reading, 1986.

Kopka, Helmut. *LATEX, eine Einführung.* 3rd edition, Addison-Wesley, München, 1992.

Kopka, Helmut. *LATEX, Erweiterungsmöglichkeiten.* 3rd edition, Addison-Wesley, München, 1991.

Nakanishi, Akira. *Writing Systems of the World.* Charles E. Tuttle Company, Tokyo, 1980.

Tonkin, Derek. *The Cambodian Alphabet.* Transvin Publications, Bangkok, 1991.