Opinions on "A graphetic approach for the Mongolian encoding model"

Jirimutu (jrmt@almas.co.jp)

Ерөнхий тойм

Overview

2017 оны 9-р сард Хөх хотод зохион байгуулсан Монгол Бичгийн Ажлын Хэсгийн хурал дээр хэлэлцэгдсэн хэлэлцүүлэгийн ерөнхий тоймыг доор дурдсан болно. Ялангуяа энд "Дүрсээр Монгол бичгийг кодчилох загвар" -ын талаархи өөрийн санал бодлоо хуваалцахыг хүсэж байна.

Following on viewpoints I have stated on WG2 Mongolian Ad Hoc meeting held in Hohhot in the September of 2017, here I elaborate my own opinions on "A Graphetic Approach for the Mongolian encoding model".

Нарийвчилсан тайлбар өгөхөөс өмнө өөрийн үзэл бодлыг дүгнэж тоймлон хүргье.

Let me summarize my opinion before giving detailed explanations.

1. "Дүрсний стандарт" –аас өмнө "Авиан стандарт" уруу хүчээр шилжих нь илүү тогтовортой, ашиглах боломжтой хэдий ч, Өвөр Монгол болон Хятад улсын 7 аймгийн нутаг дэвсгэрт Монгол хэл бичиг хэрэглэдэг хүмүүсийн хувьд Монгол хэл бичгийн хэрэглээг шахаж, эрсдэлд хүргэх болно. Улмаар энэ байдал нь Монгол хэлний хэрэглэгчдийн тоог эрс бууруулах боломжтой гэсэг үг. Кодлох стандарт нь тогтворжихоос өмнө өөрчлөгддөг ба ирээдүйд урт хугацааны явцад Монголын хэрэглэгчид вэбсайт болон мобайл сүлжээний орчинд монгол бичгийг ашиглах боломжгүй болхоос гадна, мөн монгол оюутанууд өөрсдийн эх хэл, скриптийг өнөөгийн ухаалаг төхөөрөмжийн, орчинд сурах боломжгүй болно гэсэн үг. Харамсалтай нь үүний үр дүнд кодын стандарт дээр "хасах болон өөрчлөгдөх" гэсэн тогтворгүй кодлох стандарт үүсэж байна. Бидний өмнөө тавьсан Монгол Юникодыг тогтворжуулж, нэгтгэх зорилгодоо

бид хүрэх боломжгүй болж байна гэсэн үг. Шинэ стандарт маан мэдээж хялбархан хүлээн зөвшөөрөгдөхгүй, хэрэглэгчдийн дунд ч мөн хялбархан өргөнөөр хэрэглэгдэхгүй харин бид 2000-2010 оны туршид байсан “кодгүй стандарт”-тай адил урт хугацааны туршид бид хүчин чармайж ажиллах болно. Хэрвээ хүчин чармайлт амжилтгүй болоод “Дүрсийн стандарт”-г ашиглахаар болвол Монголчуудад энэ нь тийм ч их тустай зүйл болохгүй.

A forceful switch to "graphetic standard" before "phonetic standard" become usable, stable and unified, that will induce the risks of strangling the usage of Mongolian language in Inner Mongolia and other seven provinces and areas in China. It will be causing drastic reduction of Mongolian language user group as well. The encoding standard changes before stabilize, for a long term period in the future, the Mongolian users will be not able to use Mongolian script on the web and mobile network environment smoothly, and also the Mongolian student will not be able to learn their mother tongue language and script on current smart device and environment. It is a sorrow result what we get from the unstable encoding and "chopping and changing" on the encoding standard. Our objectives for stabilizing and unifying effort for Mongolian Unicode will not be able to get reach the correct goal. The new standard will not be easily accepted and also utilized between users widely, we will come into another long term "non-encoding-standard" period like between 2000 and 2010. If the effort get opposite effect, the decision to use graphetic standard for Mongolian is not helpful for Mongolians. Maybe it will become the main reason for increasing instability of Mongolian Unicode.

2. Кодчлолын үүднээс, Монголын кодчлох стандартад дүрсээр кодлох аргыг хэрэглэх нь үнхээр зөв шийдвэр юм, харин хамгийн тохиромжтой арга нь араб хэлний товчилсон нийлбэрийн загварт суурилдаагүй, алдаа мадаггүй үнэн зөв дүрсийн стандарт дээр тулгуурлах хэрэгтэй.

   From the perspective of encoding, using graphetic approach for Mongolian encoding standard is absolutely correct, and the most reasonable approach is pure graphetic approach which is not based on Arabic cursive joining model.

3. WG2 N4889 дугаартай бичиг баримтанд Монгол текстийн ердийн хэвшмэл хэрэглээнд зориулагдаагүй олон төрлийн асуудлуудыг хөндөж харуулсан байдаг. Тэгэхээр цаашид улам сайжруулж боловсруулах үүднээс та бүхэндээ зарим асуулт болон санал хүсэлт тавих гэсэн юм.

   The Document WG2 N4889 has many unresolvable problems for usual normal applications of Mongolian text. So here let me ask some questions and give some suggestions for further improvements.

4. Хэрэв Монгол улсын кодчиллын загварт зориулж дүрсийн стандартыг боловсруулах шаардлагатай бол хүлээн зөвшөөрөгдөх хэд хэдэн стратеги боловсруулах хэрэгтэй. Нэгдүгээрт: бид өнөөгийн авиан стандартыг нэгтгэж, тогтворжуулах хэрэгтэй. Хоёрдугаарт: Бид өнөөгийн авиан стандарттай хамтран оршин тогтох чадвартай дүрмийн стандартыг боловсруулах хэрэгтэй. Эцэст нь: Бид дүрсийн стандартыг аажмаар хэрэгжүүлж авиан стандартыг хэрэглэхээс эсэргүүцэх хэрэгтэй. Ирээдүйд, авиан стандарт нь зөвхөн Монгол хэл дээрхи хэл шинжлэл, судалгааны туслах стандарт болж хувирна.

   If it is a must to design an ideal graphetic standard for Mongolian encoding model, the adoptable strategy should be: firstly, we should unify and stabilize current phonetic standard; secondly, we have to design a graphetic standard which can co-exist and interchange with current phonetic standard; finally, we need to gradually enforce the graphetic standard, and deprecate the phonetic standard. In the future, the phonetic standard will be used solely as auxiliary standard of Mongolian for linguistic research.

5. Хүмүүст өөрийн бодол санааг ойлгомжтой болгохын тулд жагсаасан зүйлсийг нэг нэгээр нь нарийвчлан тайлбарлаж өгье.

   In order to let everyone clearly understand my opinions, let me explain the listed items in detail one by one.

   >>>>

1. The risk of forceful switch to graphetic standard before stabilization of current Mongolian encoding phonetic standard

The current Unicode standard of Mongolian, i.e. called as phonetic standard, was designed and proposed under the circumstance of not having a usable national standard. Before the standard come to Unicode and implemented on the system, the standalone and web application of Mongolian was widely used in Inner Mongolia, but all of the vendors use their own independent encoding systems on that time.

Since the Mongolian encoding standard, i.e. current phonetic standard was accepted by ISO 10646 in 1999 and included into UNICODE 3.0 which was published in 2000, we had been expecting to use the new encoding standard in the application early time. But just because there was no OS level support for phonetic Mongolian encoding standard using Arabic cursive joining model at that time, we continued almost another 10 years of having no standard for Mongolian in Inner Mongolia.

Starting from the release of Windows Vista in 2007 which provides first Mongolian OpenType font as well as the implementation technique and the rendering engine, we began to utilize the Mongolian Unicode standard. It is after 2010, that a lot of local vendors started to switch to Unicode Mongolian in their application systems. In other words, the Mongolian font and rendering engine provided by Windows 7 satisfied the basic requirements of users.

In the past 10 year's effort on R&D and market promotion, we have received generous financial support from the state and government of autonomous region and other 7 regions and areas of China which use Mongolian, we have carried out the situation that Unicode Mongolian is utilized in whole area on all of the Mongolian systems and applications (except some of legacy application). We are coming to new situation of all Mongolian applications and systems support Unicode Mongolian encoding throughout industries of technology, culture, education, entertainment, sports, etc. Although current phonetic standard has many problems, but besides one or two significant problems, compared to not having a standard at all, some other problems are negligible. Their scopes of effect are very little. Our users largely accept and bear with them currently. Of course, as users, they have desire for further improvements and perfections of our encoding standard.

If hastily decide to change to graphetic standard before stabilization and unification of current widely used Mongolian Unicode standard, the effect is non-negligible. It is a very dangerous decision for "chopping and changing" the encoding standard which may well induce drastic reduction of Mongolian script usage and users. It will end the possibility of broadly using Mongolian script Inner Mongolia and other 7 provinces and regions in China.

The main reasons are following in the list：

- Mongolian script will fall back into period of not having an encoding standard.
- At least 5-10 years are required to reach current situation of widespread usage for a new Mongolian Unicode standard which is going through its design, release, implementation, and promotion as well as marketing period.
- The government gave big financial support for promotion of current standard as well as digitization of Mongolian script, if what had been done before was wrong, and must start new R&D and promotion for new graphetic standard, that huge financial support is no longer available.
- Vendors will lose interest in "chopping and changing" the encoding standard, and then abandon continuous R&D and promotion.
- Users will find the encoding standard switching is painful and will become objectionable, unsatisfactory and furious. They can only discard it out of frustration.
- It will probably cause confusion among children and let them abandon Mongolian language. For quickly fit to the social and environment changes, and they will switch to learning Chinese, greatly decrease the desire and passion of children for learning Mongolian language and script.
- In nowadays, our daily life, study and work are inseparable from computers, mobile phones and networks. Users will be confused to not having usable Mongolian encoding standard, and temporally come into the situations that they do not want to use Mongolian script on the computers, mobile phones and networks. Even they want to use Mongolian, but there are no plenty of available, stable Mongolian systems caused by the encoding standard switch impact. They can only use Chinese as network communication language on that time in China.
- If normal users cannot use Mongolian script on internet to communicate, the teachers cannot use Mongolian script on their

computer, or even not teaching children to learn their own Mongolian language and script using computers, and also If the situation continues for another 5-10 years or even more longer, that will deprive one generation's possibility of using Mongolian language and script, which cause drastic decline of users of Mongolian language and script, and even gradually progress towards extinction.

一、 From the perspectives of encoding principles, encoding for Mongolian should use pure graphetic approach

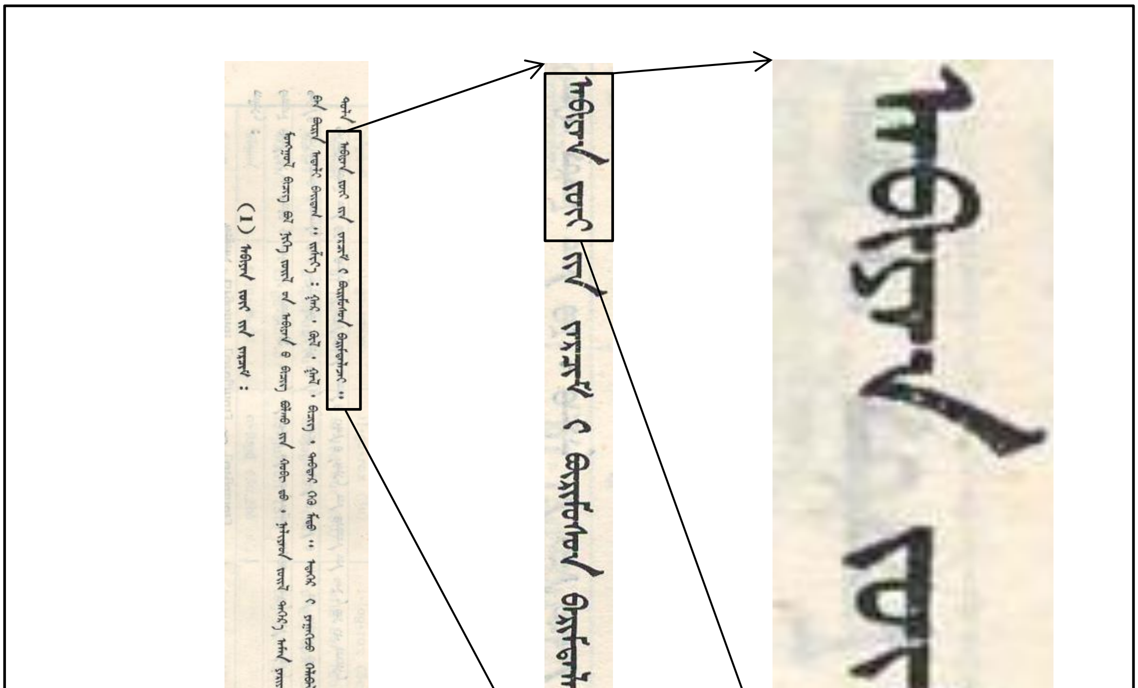
1. Historically, Mongolian printing methods all use glyphs as minimal unit

Mongolian printing industry had undergone historical periods of woodcut printing, metal font type printing, typewriter and electronic typewriter, computer printing and desktop typeset publishing system etc. During those periods, different kinds of font glyphs were used as minimal printing unit of Mongolian. Let us see glyphs used prior to digitization.

●Metal font type

Until now, no pictures of real physical metal font type publication layouts of Mongolian books have been found yet. Picture 1 listed below is a sample part of metal font type Mongolian book.

The trace of every metal font type glyph can be found in picture 1. For Example, the glyphs of last 2 words (the rightmost vertical line) in the following picture are: ᠬᠤ ᠪᠢ ᠯᠭᠠ ᠨ ᠳᠤ ᠷ are all independent glyphs. It can be seen that Mongolian syllables were used as the minimum unit font for the efficiency of typesetting.

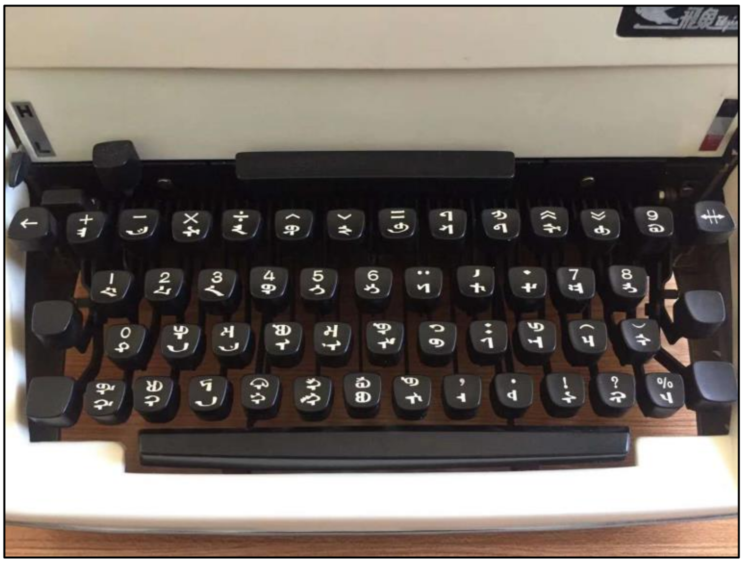


Picture 1 metal font type Mongolian sample

●Typewriter

Picture 2 listed below is a Mongolian typewriter we found. It's keyboard layout is shown in picture 3. The real typewriter is stored in Delehi Information Technology Co. Ltd.

Picture 2 Mongolian typewriter



Picture 3 Mongolian typewriter keyboard layout

It can be seen from picture 3 that, many Mongolian component glyphs are used as minimal glyph units, because of the limited number of keys on a typewriter.

2. The Mongolian encodings on computer systems are all graphetic

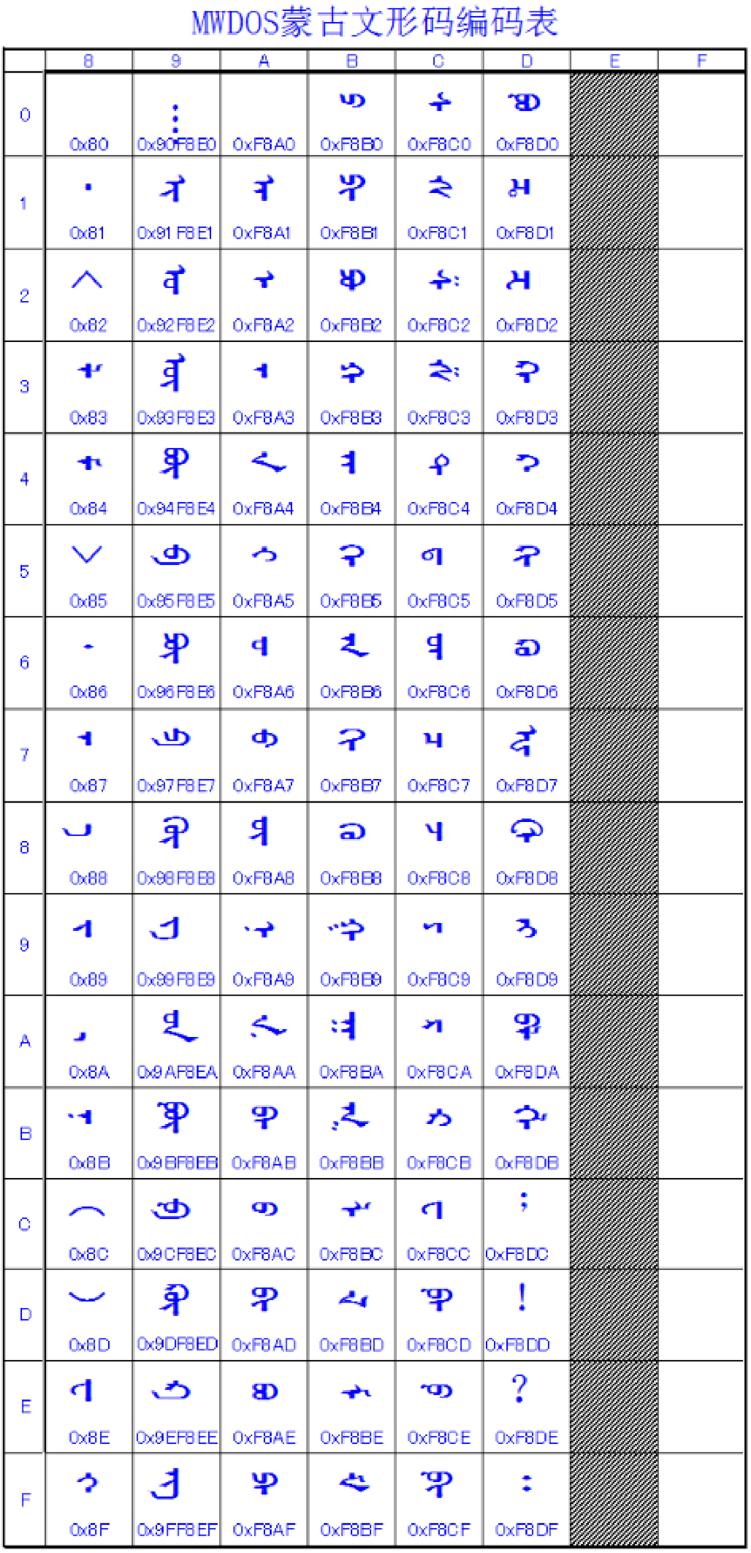
Many Mongolian application systems had been used since the beginning of Mongolian input, display and printing on computers in 1980s. Those systems all use font glyphs as encoding unit without exceptions, i.e. graphetic encoding. For example: the following systems listed below are all Mongolian graphetic encoding system.

- ✧ Earlier MWDOS system
- ✧ Hua Guang typesetting and publishing system
- ✧ Fang Zheng typesetting and publishing system
- ✧ MenkSoft 2002 system
- ✧ Saiyin encoding system, etc.

And most of these systems encode graphically for all variant forms of characters and ligatures. Only MenkSoft 2002 completely encoded for all variations forms of the glyph, and Mongolian ligatures are also encoded as two separated character glyphs joined together.

One important point is: these systems all encode vowels ᠡ ᠠ ᠢ ᠣ ᠥ independently. It is because these vowels are handled as minimal unit for nearly a century in publishing industry, besides of the limitation of the key count on the typewriters. For this reason, we can't accept that from the emotional perspective considering cultural heritage, if these minimal units of the Mongolian vowels are overlooked.

If Mongolian glyph component must be used for Unicode encoding in order to control the number of code points, that must be the solution using pure graphetic encoding for Mongolian glyphs. Once the initial, medial and final positional variant forms of Mongolian glyph components are introduced in the encoding, it will distort Mongolian inherent rules. It is rather difficult for Mongolian people to correctly use this encoding depend on their knowledge of Mongolian language and script. This is the main reason why all Mongolian people rejected the N4889 proposed approach on WG2 Mongolian Ad Hoc meeting in September 2017. Including me, it is really hard to accept the proposed encoding approach based on 3 positional variation forms of Mongolian glyph components.



MWDOS蒙古文形码编码表

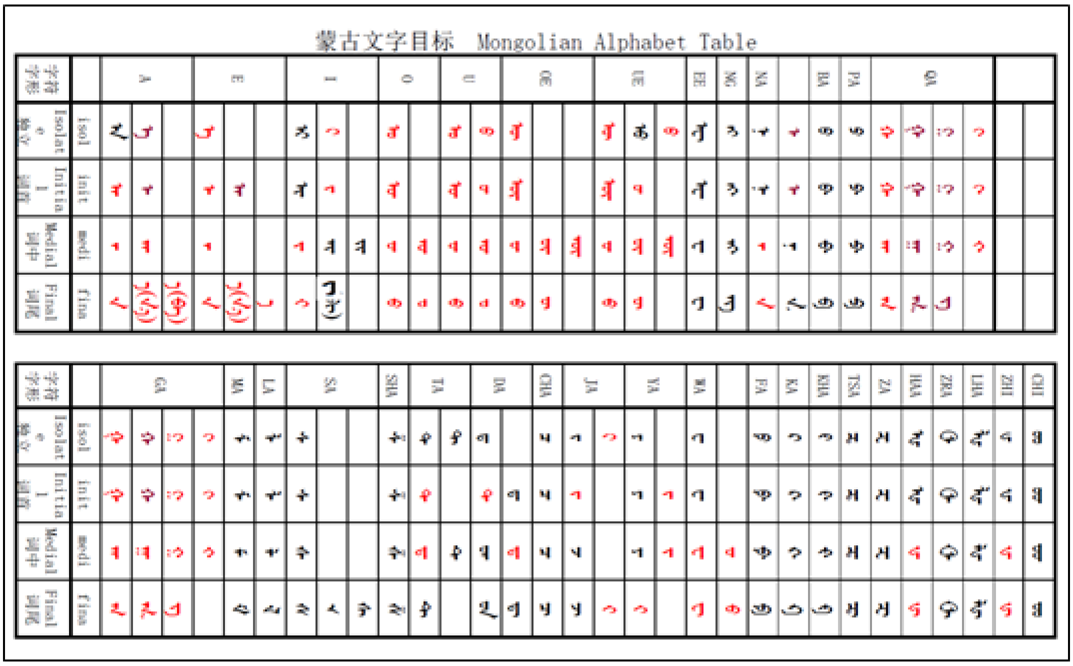
	8	9	A	B	C	D	E	F
0	0x80	0x90F8E0	0xF8A0	0xF8B0	0xF8C0	0xF8D0	▨	
1	0x81	0x91 F8E1	0xF8A1	0xF8B1	0xF8C1	0xF8D1	▨	
2	0x82	0x92F8E2	0xF8A2	0xF8B2	0xF8C2	0xF8D2	▨	
3	0x83	0x93F8E3	0xF8A3	0xF8B3	0xF8C3	0xF8D3	▨	
4	0x84	0x94F8E4	0xF8A4	0xF8B4	0xF8C4	0xF8D4	▨	
5	0x85	0x95F8E5	0xF8A5	0xF8B5	0xF8C5	0xF8D5	▨	
6	0x86	0x96F8E6	0xF8A6	0xF8B6	0xF8C6	0xF8D6	▨	
7	0x87	0x97F8E7	0xF8A7	0xF8B7	0xF8C7	0xF8D7	▨	
8	0x88	0x98F8E8	0xF8A8	0xF8B8	0xF8C8	0xF8D8	▨	
9	0x89	0x99F8E9	0xF8A9	0xF8B9	0xF8C9	0xF8D9	▨	
A	0x8A	0x9AF8EA	0xF8AA	0xF8BA	0xF8CA	0xF8DA	▨	
B	0x8B	0x9BF8EB	0xF8AB	0xF8BB	0xF8CB	0xF8DB	▨	
C	0x8C	0x9CF8EC	0xF8AC	0xF8BC	0xF8CC	0xF8DC	▨	
D	0x8D	0x9DF8ED	0xF8AD	0xF8BD	0xF8CD	0xF8DD	▨	
E	0x8E	0x9EF8EE	0xF8AE	0xF8BE	0xF8CE	0xF8DE	▨	
F	0x8F	0x9FF8EF	0xF8AF	0xF8BF	0xF8CF	0xF8DF	▨	

Picture 4 earlier encoding table for MWDOS

3、Issues of Mongolian graphetic encoding under the Arabic cursive joining model

(1) The unique features of Mongolian variant form relation

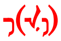
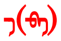
Although Mongolian cursive joining feature is inherited from Arabic, as we know the Arabic cursive model, one Arabic character has 3 positional variant forms and only has one unique variant form on each position so far. But compared to the Arabic, one Mongolian character has more variant forms on one position and also different characters have same variant form glyph on some position. Mongolian variant form feature is too complicated that we cannot simply use Arabic cursive model to satisfy all of these complicated Mongolian variant form feature requirements.


蒙古文字目标 Mongolian Alphabet Table
字符
字形
Isolat 独立 isol
Initia l 词首 init
Medial 词中 medi
Final 词尾 fina
A
E
I
O
U
OE
UE
EE
NG
NA
BA
PA
QA
GA
MA
LA
SA
SHA
TA
DA
CHA
JA
YA
RA
FA
KA
KHA
TSA
ZA
HAA
ZRA
LHA
ZHI
CHI

Picture 5 Table of Mongolian alphabet Variants

As illustrated in Picture 5 listed above, those red variant forms are shared variant form glyph of different characters (alphabets). The dark red colored variant forms can be excluded from the table. It is the group of special variant forms defined by linguists, but it is not the part of regular alphabet table.

For example: following listed variant forms are all shared among different characters.

variant forms	character variation name symbol
	A-isol,E-isol;
	A-init,E-init;
	E-init,A-Init;
	A-medi,E-medi,NA-medi;
	A-medi,QA-medi,GA-medi;
	A-fina,E-fina,NA-fina;
	A-fina,E-fina;
	A-fina,E-fina;
	I-medi,YA-medi;
	I-fina,JA-fina,YA-fina;
	I-isol,JA-isol;
	I-init,YA-init;
	O-isol,U-isol;
	O-init,U-init;
	U-init,UE-init
	O-medi,U-medi,OE-medi,UE-medi,WA-medi;
	O-medi,Umedi;
	O-fina,U-fina,OE-fina,UE-fina,WA-fina;
	O-fina,U-fina;
	U-isol,UE-isol;
	OE-isol,UE-isol;
	OE-init,UE-init;
	OE-medi,UE-medi;
	OE-medi,UE-medi;
	OE-fina,UE-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;
	QA-fina,GA-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;
	QA-medi,GA-medi;
	QA-fina,GA-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;

	QA-medi,GA-medi;
	QA-fina,GA-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;
	QA-medi,GA-medi;
	TA-init,DA-init;
	TA-medi,DA-medi;
	EE-medi,WA-medi;
	EE-fina,WA-fina;
	HAA-medi,ZHI-medi;
	HAA-fina,ZHI-fina;

(2) Problems caused by Mongolian encoding based on Arabic cursive joining model

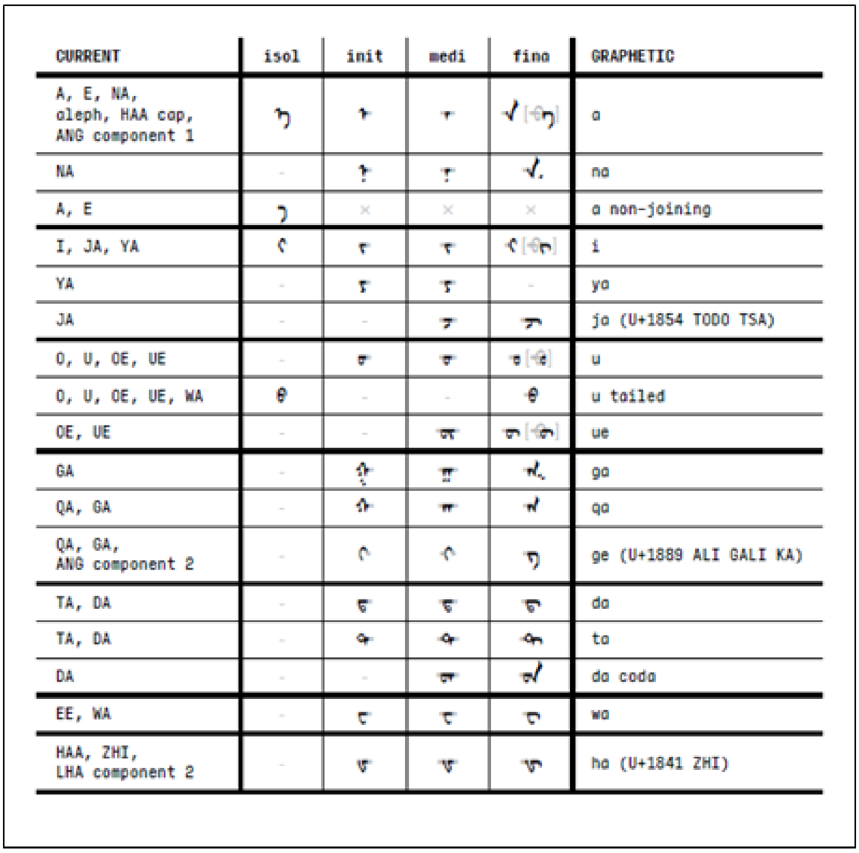
(2.1) One-form-multiple-code problem of phonetic encoding

Using Arabic cursive joining model for Mongolian encoding, there will be many initial, medial and final forms of different characters have same variant form glyphs. It will causes phenomenon of multiple codes for same visual forms. This is widely known problem of the phonetic encoding. The Improvement approaches for this problem will not be discussed here. Please find separated documents for the improvement proposals.

(2.2) String manipulation issue of graphetic encoding

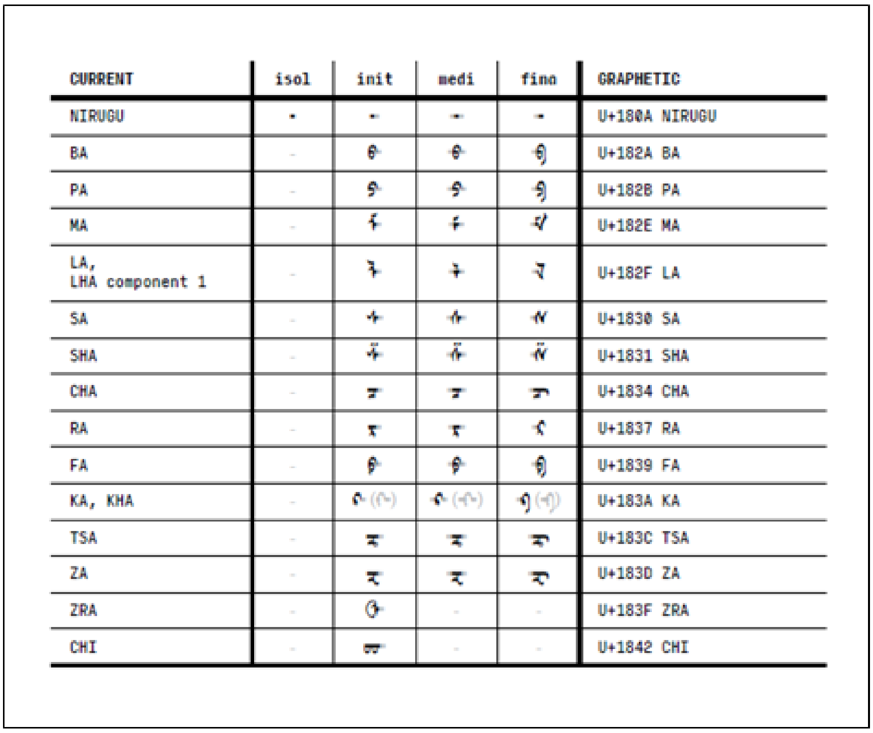
Using Arabic cursive joining model is tearing off relations between variant forms and alphabet characters of Mongolian. As long as one character (alphabet character or component) exists only one variant form on each of 3 different positions, the encoding code point will become the "nominal character". The initial, median and final presentation form of the "nominal character" will be all unique, like WG2 N4889. I am not preparing to talk about whether the selection of the "nominal character" and the representation form of WG2 N4889 is reasonable or not now. Firstly let us come to see how the string manipulation issue exists on this encoding approach.

For referential simplicity, I listed the copies of the encoding table from document N4889 in picture 5 and 6.



CURRENT	isol	init	medi	fina	GRAPHETIC
A, E, NA, aleph, HAA cap, ANG component 1					a
NA	-				na
A, E		×	×	×	a non-joining
I, JA, YA					i
YA	-			-	ya
JA	-	-			ja (U+1854 TODO TSA)
O, U, OE, UE	-				u
O, U, OE, UE, WA		-	-		u tailed
OE, UE	-	-			ue
GA	-				ga
QA, GA	-				qa
QA, GA, ANG component 2	-				ge (U+1889 ALI GALI KA)
TA, DA	-				da
TA, DA	-				ta
DA	-	-			da coda
EE, WA	-				wa
HAA, ZHI, LHA component 2	-				ha (U+1841 ZHI)

Picture 6 first half of encoding table from doc. N4889



CURRENT	isol	init	medi	fina	GRAPHETIC
NIRUGU					U+180A NIRUGU
BA	-				U+182A BA
PA	-				U+182B PA
MA	-				U+182E MA
LA, LHA component 1	-				U+182F LA
SA	-				U+1830 SA
SHA	-				U+1831 SHA
CHA	-				U+1834 CHA
RA	-				U+1837 RA
FA	-				U+1839 FA
KA, KHA	-				U+183A KA
TSA	-				U+183C TSA
ZA	-				U+183D ZA
ZRA	-		-	-	U+183F ZRA
CHI	-		-	-	U+1842 CHI

Picture 7 second half of encoding table from doc. N4889

Because all members meeting here are experts on encoding, I will skip talking about what is string manipulation. Here I only want to discuss whether N4889 encoding approach can satisfy users after following 2 kind of string manupulation.

(1) Will concatenating two strings result in correct Mongolian text? Will split out the characters with first string's length from the beginning of the string result in same strings before joining?

Firstly, Let us see what is the result getting from the string manupulation on other languages.

English：rain + bow => rainbow => split(4)= rain + bow；

Chinese：人民+公社 => 人民公社 => split (2) = 人民 + 公社

Japanese：お困り+でした => お困りでした => split (3) = お困り+でした

Now let us see what happens for Mongolian text under N4889 encoding.

(a) ᠠᠠ + ᠠᠢᠳᠠᠰ = ? What kind of string should we get?

（Mongolians will answer that should results in ᠠᠠᠠᠢᠳᠠᠰ）

But, what is the result under N4889 encoding?

ᠠᠠ (a+a) + ᠠᠢᠳᠠᠰ (a+i+da+a+sa) = ᠠᠠᠠᠢᠳᠠᠰ(a+a+a+i+da+a+sa)

Different from expectation, but come out somewhat close.

(b) ᠴᠢᠠᠭ + ᠢᠤᠷᠢᠬ = ? What kind of string should we get?

（Mongolians will answer that should result in ᠴᠢᠠᠭᠢᠤᠷᠢᠬ）

But, what is the result under N4889 encoding?

ᠴᠢᠠᠭ (ca+i+a+ge) + ᠢᠤᠷᠢᠬ (i+u+ra+i+qa) = ᠴᠢᠠᠭᠢᠤᠷᠢᠬ(ca+i+a+ge+i+u+ra+i+qa)

unexpected, unimaginable!!!

(c) ᠰᠢᠷᠤᠢ + ᠠᠷᠳᠠᠨᠢ = ? What kind of string should we get?

（Mongolians will answer that should result in ᠰᠢᠷᠤᠢᠠᠷᠳᠠᠨᠢ）

But, what is the result under N4889 encoding?

ᠰᠢᠷᠤᠢ (sa+i+ra+u+i) + ᠠᠷᠳᠠᠨᠢ (a+ra+da+a+na+i) =

ᠰᠢᠷᠤᠢᠠᠷᠳᠠᠨᠢ (sa+i+ra+u+i+ a+ra+da+a+na+i) Mongolians read this as (siruerdeni)

Hugely different from our expectations, so unacceptable for Mongolians!

(c) ᠰᠤᠳᠤ + ᠯᠤᠯ = ? What kind of string should we get?

（Mongolians will answer that should result in ᠰᠤᠳᠤᠯᠤᠯ）

But, what is the result under N4889 encoding?

ᠰᠤᠳᠤ (sa+u+da+u-tailed) + ᠯᠤᠯ (la+u+la) =

ᠰᠤᠳᠤᠯᠤᠯ(sa+u+da+u-tailed+la+u+la)， presentation form for u-tailed is undefined!

This is not the form we want to see. Normally, we do not want the final form appears in the middle of a word!

Of course, tokenizing on lengths of comprising substrings will definitely result in original substrings.

In order to solve these mismatch of real and expected composites, the method of introducing smart correction for composite words, will be more troublesome and complex.

Although I know the method of introducing smart correction is infeasible, I want to discuss one example applying the author's hypothesis of introducing smart editor software which assist in correction for string manipulation (concatenating and tokenizing).

Suppose a Mongolian word (string) ᠠᠨᠠᠷᠢᠯ, tokenize it, smart correct, concatenate again, unclear how to smart correct the joined string.

ᠠᠨᠠᠷᠢᠯ (a+na+a+ra+i+l) => ᠠᠨ(a+na)+ ᠠᠷᠢᠯ(a+ra+i+l) => ᠠᠠ(a+a)+ ᠠᠷᠢᠯ(a+ra+i+l)

ᠠᠠ(a+a)+ ᠠᠷᠢᠯ(a+ra+i+l) => ᠠᠨᠠᠷᠢᠯ (a+a+a+ra+i+l) (anril) ?

ᠠᠠ(a+a)+ ᠠᠷᠢᠯ(a+ra+i+l) => ᠠᠨᠠᠷᠢᠯ (a+na+a+ra+i+l) (eneril) ?

So it is impossible to use smart spell check and editing system to guarantee the correctness of string for string manipulation and edit. Smart correction for N4889 encoding is only usable for inputting stage, but more troublesome for editing stage, or unhelpful at all.

⑵ What are the presentation forms of substring after tokenization? Will rejoining after tokenizing result in same original string?

Firstly, Let us see what is the result getting from the string manupulation on other languages.

English：speedup => split(5)=> speed + up => speedup；

Chinese：联合作业 => split (2) => 联合+作业 =>联合作业；

Japanese: 迷惑をかける => split (3) =迷惑を+かける =>迷惑をかける;

Now see what happens for Mongolian text under N4889 encoding.

(a) [illegible] => split(5)=> ？ Mongolians will expect that should result in ([illegible] + [illegible])

But in reality, that result in [illegible] + [illegible]， Different from expectation!

=> [illegible] => split(6)=> ？ Mongolians will expect that should result in ([illegible]

+[illegible])

But in reality, that result in [illegible] + [illegible], Split on wrong point!

Rejoining the tokenized substring, of course we can get correct original string. => [illegible] , Sure, that is the result not using smart correction methods.

(b) [illegible] => split(3)=> ? Mongolians will expect that should result in ([illegible] + [illegible])

But in reality, that result in[illegible] + [illegible], Still different from expectation!

(c) If I wanted to input compound word ᠠᠭᠠᠰᠢ ᠵᠠᠩ, using currently familiar phonetic code (agasi jang), but because of the space in the middle of the words is not being pressed well enough to recognize on assumption, and it becomes (agasijang). According to smart conversion from phonetic code to N4889 encoding, it should be ᠠᠭᠠᠰᠢᠵᠠᠩ, when realized the space was absent and move the cursor in front of the character ᠵ, then add a space, what will happen?

ᠠᠭᠠᠰᠢᠵᠠᠩ => split(6)=> ? Mongolians will expect that should result in (ᠠᠭᠠᠰᠢ ᠵᠠᠩ)

But in reality, that result in ᠠᠭᠠᠰᠢ ‍ᠵᠠᠩ, users do not know how to convert ᠵ to initial ᠵ ?

When it is become users must for users to retype ᠵᠠᠩ, and relaying on the function which IME will help for them to input and correct the word. For this users will say No!.

(2.3) N4889's graphetic approach does not solve one-form-multiple-code problem entirely in the fact.

(a) ᠬ (medial qa) and ᠠᠠ (medial a+a) are same

(b) ᠦ(medial ue) and ᠣᠢ (medial u+i) are same

(c) ᠦ (medial ue) and ᠣᠠ (final u+a) are same

(d) **ᠴ (initial CHI) and ᠣᠣ (initial u+u) are same**

(2.4) Best option for graphetic encoding approach for Mongolian is pure graphetic approach

From the perspectives of the mentioned issues above, both phonetic and graphetic encoding approach using Arabic cursive joining model for Mongolian have their own disadvantages. It is caused by the inherent features of Mongolian script itself and is non-negligible for encoding approach.

If you say the graphetic encoding is desirable and also want to solve all of the problems above, using pure graphetic encoding approach is the best option. Furthermore, there is experience of using the pure graphetic encoding approach for many years in Inner Mongolia. The ideal choice is fully analyzing the advantages and disadvantages learned in the past and designing a solution which is feasible and easily acceptable to the majority of Mongolians.

4、 Questions for some issues of WG2 N4889 graphetic approach

(1) Three most important Mongolian initial Alphabets (A、E、I) do not have independent code points in N4889

Firstly, let me talk from the perspectives of all Mongolian people who understand their own language and script, the N4889 is an encoding approach which is splitting the character till the variant form components in fact. In the history, we can find some articles recorded like this kind of explanation, but this method is not widely used in current Mongolian linguistic education. We were never taught about this kind of explanations in the school. For this reason, the majority of Mongolians are unfamiliar with this concept. Therefore, this kind of encoding approach may induce the risk of another Mongolian written script reformation.

Based on our language and literature knowledge we have learned since childhood, as well as contemplating nearly 40 years of digitization experience, at least three initial alphabets ᠠ ᠡ ᠢ or ᠠ ᠡ ᠢ (A、 E、 I) of Mongolian should be independently defined into code points in the Mongolian encoding standard. It is the most basic requirement for Mongolian people like any other languages done.

Although N4889 can display all Mongolian vowel's variant forms using the code point component, but there are only one E and half I, without A in the code point table is hardly acceptable to Mongolian people. In the torrents of history, If a language does not distinguish A and E, that is the biggest crisis. Japanese is a typical example. I am living in Japan and know that Japanese does not have E vowel in their language. Japanese people, if they have no experience living or learning outside Japan, they cannot easily or at all distinguish A and E. they pronounce both of them as A(あ). It is causing the biggest barrier for foreign language learning in Japan now as I know.

Mongolian is one of a few languages which are rich of vowels. Native 7 vowels plus loan vowel ᠧ （EE）, 8 vowels in total. This count does not include half vowel ᠶ、ᠸ（YA、WA）which is labeled by some linguists. We concern that N4889 encoding approach will bury this advantage of Mongolian language deep in the encoding level of the language, maybe it will induce that some of the most important Mongolian vowels will disappear someday in future. If we have no symbols to record the pronounced vowel, the vowel will have risk of disappearing from the language as I understand. Although I do not know Japanese language evolvement history, I always think there must be a historical period or people who pronounce E vowel, merely because there is no character designate to record that pronunciation, during the long period of education, maybe Japanese people lost their E pronunciation. They are naturally pronouncing "Earth" as "アース(same pronunciation with arth)", causing incomprehension of people from English world（Excuse me, I am not criticizing Japanese language. But to only state the meaning of taking written characters for vowel pronunciations seriously. Compared with our Mongolian language, Japanese is much more advanced in other all aspects, we are always learning advanced concepts and technology from them）.

(2) Should assign variant forms for nonexistent isolated forms as well as initial, medial and final forms.

In N4889, there are no variant forms defined for nonexistent isolated forms as well as initial, medial and final forms of character (or code point). According to Arabic cursive joining model, this kind of definition for invalid characters are unacceptable.

Firstly, every character（Unicode code point）is possibly appears as isolated position or in the initial, medial and final position of the word. Secondly, when those invalid variant forms appeared in the word, neither using alternative character nor leaving as blank is not reasonable. No one can distinguish the underlying codes of them. It is another kind of "security issue" of the encoding approach in fact!

(3) Text inputting and editing issue

(a) Text inputting issue

Firstly, when introducing N4889 encoding approach on WG2 Mongolian Ad Hoc meeting, it was stated that user experience will not be affected when inputting Mongolian. But, what we have noticed from our detailed examination, the encoding approach will be not affecting user experience is too stretching. In N4889, MVS, NNBSP and FVS1-3 are completely abandoned. So I want to ask that how to implement isolated A without using auxiliary key-button such as MVS? Following the same user experience with current input method, would you please provide input sequences for the following 2 words.

ᠰᠠᠷᠠ (sara？) save as ᠰ(sa) + ᠠ(s) + ᠷ(ra) + ᠠ(a)

ᠰᠠᠷ᠎ᠠ (sara？ sar<MVS>a？) save as ᠰ(sa) + ᠠ(s) + ᠷ(ra) + ᠠ(a non-joining)

(b) Text editing issue

In previous "string manipulation issue" section, I have already discussed the text editing issue of N4889 encoding approach. The results users can get from text editing are completely different from what they expected based on their Mongolian language and literature knowledge. Therefore, how can we say that users do not have to think about real code behind graphetic encoding, just easily following their already acquired knowledge to use it?

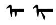
In addition, before users fully understand N4889 encoding concept, text editing efficiency is not the same as before. It is totally impossible to precisely correct wrong characters in a word.

When I am debating with N4889's authors about the editing issue, he says that it is possible for some system that we can re-acquire the word context under cursor to IME, we can use IME smart word correction logic to handle the editing issue. But I know that some of Windows applications provide this kind of functionalities. But it is unknown whether or not that other systems all support such functionalities or not? We cannot afford to wait once more time for all systems support these technologies and functionalities. We have to depend on existing technologies and system functionalities to effectively utilize our encoding standard!

The unpredictable, inconvenience issue of N4889 Mongolian encoding approach will greatly impact to user experience and decreases the work efficiency.

(4) Uncertainty of character boundary issue

N4889 encoding approach uses Mongolian variant form component as the minimal unit (code point). Hence, many Mongolian characters (alphabets)

need to be presented by 2 or more code points combination. For example: 
...... all need 2 or more code points to present them. It is become possible to decompose by code point unit, even those vowel variant form in Mongolian are non-dividable independent part. It is unreasonable to divide or split this vowel variant form. Is there any need to explicitly point out the "character boundary"? But N4889 encoding approach has not explicitly defined how to decide the "character boundary" in the text.

Could anyone please explain how other languages in Unicode are defining character boundaries? Is it better or not to request N4889 Mongolian encoding approach provide the "character boundary" logic?

(5) String manipulation issue

The string manipulation issue of N4889 has already been discussed in the previous section. I will not repeat it here.

(6) Details of property definition of "a non-joining"

We found that all initial, median and final variant forms of "a non-joining" are all "x" in N4889 encoding approach. Would you please explain the Unicode properties of this character (code point)?

The font rendering engine how to determine the previous character of this "a non-joining" should be final form, but not medial form?

Does the rendering engine do this decision and transfer to font feature? Where can I find the property definition files for all Unicode characters?

Whether or not it can be seen as that only "a non-joining" character is "pure graphetic code point" in the N4889 Mongolian code point? Shall we select all Mongolian code point as "pure graphetic code point"?

What is the difference between marks "-" and "X" in N4889 encoding table?

(7) Coexistence with current standard issue

N4889 encoding approach does not provide the research on the feasibility of coexistence with current Mongolian encoding standard. The new approach will not be accepted by users if it cannot be used in parallel with current Mongolian encoding standard.

(8) Two-way conversion with current standard issue

N4889 encoding approach does not provide research on the feasibility of two-way conversion with current Mongolian encoding standard. The promotion of new standard will face huge resistance if it will not support interchange with current Mongolian encoding standard.

5、**The suggestion for the deal steps and goals of Mongolian graphetic standard**

If the decision is made to enforce **Mongolian graphetic standard, the ideal steps and goals should we take are:**

Firstly, stabilize and unify current Mongolian encoding standard as soon as possible.

Second, design ideal Mongolian encoding standard with graphetic approach to comply with coexistence and interchangeability between new encoding standard and current encoding standard.

Finally, progressively adopt new Mongolian graphetic encoding standard, and deprecate current Mongolian phonetic encoding standard.

In future, Mongolian "phonetic encoding standard" can be used as auxiliary encoding approach for linguistic research.

In conclusion, we are all hoping our Mongolian language have a complete, stable, unified and ideal Unicode standard as soon as possible. But it is preconditioned on no big impact to current widely using standard first. Clever design， smooth implementation, parallel utilization, progressive exchange is necessary for the Mongolian user's acceptance..

The Opinion of the document is agreed with：Siqin Bilige(siqin@almas.co.jp), Bao Haishan(baohaishan@delehi.com), Burigudu(burgood@vip.sina.com) etc.

关于《蒙古文编码形码方案》的意见

吉日木图（jrmt@almas.co.jp）

概述

相继2017年9月份在呼和浩特市举行的WG2蒙古文专题会议上阐述的观点，我在此详细阐述我个人对《蒙古文编码形码方案》的观点。

给出详细解释之前，先给出我想说的观点的总结。

1. 如果没有把现行的“音码方案”标准达到可用、稳定与统一之前，强行转向采用“形码方案”标准的决定，具有阻碍蒙古文在内蒙古自治区以及中国八省区广泛使用的可能性，导致蒙古文用户群体急剧缩小的风险。在将来的很长一段时间内，用户无法在当前网络环境中顺畅使用蒙古文，学生无法在先进设备上学习自己的母语。这是一个很悲哀的结果。为了蒙古文编码的统一与稳定而做的努力，达不到尽快应用的目标，而进入长期无法接受和使用的状态，获得“适得其反”的效果的话，这样的决定对我们不但没有帮助，反而转变为加剧不稳定因素的根源。
2. 从编码原则考虑的话，蒙古文编码标准采用“形码方案”是绝对正确的，并且最合理的方案是不基于“阿拉伯连写模型”的纯粹的“形码方案”。
3. WG2 N4889 的方案存在很多不能解决的蒙古文正常应用的问题。为此提出了很多质疑点与完善建议。
4. 如果一定要为蒙古文编码设计一个理想化的“形码方案”标准的话，应该采用的策略路线是：首先、尽快把当前“音码方案”标准的统一与稳定；其次、设计与现行“音码方案”标准共存并能互转的“形码方案”标准；最后逐渐启用“形码方案”标准，废弃“音码方案” 标准；将来“音码方案” 标准为语言学研究等作为辅助编码方案。

为了让大家更清楚了解我的观点，下面对上述的观点进行逐一详细解释。

二、 现在的蒙古文编码方案稳定之前强行决定采用“形码方案”标准的危险性

目前的蒙古文的国际标准，即“音码方案”标准是在没有一个实际可用的现行国家标准的情况下直接研究设计与提案形成的标准。当时蒙古文的信息化应用也很广，但是各厂家都采用各自独立的编码方案。

1999 年蒙古文编码标准，即现在的“音码方案”被 ISO 10646 接受，编入 2000 年出版的 UNICODE 3.0 以来，由于当时还不存在系统级支持该编码方案的技术而蒙古文持续了将近 10 年之久的没有编码标准的现状。

自从 2007 年的 Windows Vista 提供了第一版蒙古文 OpenType 字体以及实现技术以及渲染引擎，才开始了蒙古文国际标准化应用的步伐。本地企业进入采用国际

标准编码是在2010年之后，也就是Windows 7 提供的蒙古文字体以及渲染引擎能够满足用户使用的初级要求。

经过将近10年的积极研发与大力推广等努力以及国家与自治区政府的大力资金支持的情况下，在中国使用蒙古文的 8 个省区，全面展开了使用蒙古文国际标准的热潮，并已经在科技、文化、教育、文艺、体育等各个领域，出现广泛应用国际标准蒙古文系统的新局面。虽然现在的“音码方案”国际标准存在很多问题，但是除了几个大问题之外，其他一些问题与没有编码标准比起来都是微不足道的，其影响范围很小。我们的用户很大程度上是能够接受和容忍的。当然作为用户，针对我们的文字方案也有进一步改进与更加完善的愿望。

如果目前已经大量使用起来了的蒙古文编码国际标准还没有达到比较稳定和统一的情况下，仓促决定改变为采用“形码方案”的国际标准，其影响是不可忽视的。这是一个有可能造成蒙古文字应用范围急剧缩小，使用用户数急剧缩小的极具危险性的决定。这个决定会扼杀蒙古文在内蒙古自治区以及中国八省区广泛使用的可能性。其理由如下：

- 蒙古文再度进入没有编码标准的时期。
- 新的标准的出台、实现、推广与达到现在这样大量应用的局面至少需要 5-10 年。
- 政府为现在的标准推广以及蒙古文信息化给予了大力的资金支持，如果说以前做的都错了，必须用新的“形码方案”编码标准，再来一次新的研发与推广，再一次大量的资金支持是不可行。
- 企业对变来变去的编码标准失去信心，放弃坚持研发与推广。
- 用户对变化无常的蒙古文编码标准产生反感、不满、愤怒，在无可奈何的情况下只能放弃。
- 有可能导致孩子们对学习自己母语产生迷惑而导致放弃学习蒙古文。为快速适应社会和环境的变化，无奈改学或改用中文，将会大大削弱了学习使用蒙古文的欲望和热情。
- 在当今社会，人们在日常生活、学习与工作都离不开计算机、手机与网的情况下，没有可用蒙古文编码标准的情况下用户会进入迷茫状态，有可能进入暂时在计算机、手机与网络上不用蒙古文的状态。因为想用也没有可用的稳定的蒙古文系统。只能使用中文作为网络交流语言。
- 在当今网络环境一般用户不用蒙古文了，老师不用蒙古文了，孩子就更谈不上用蒙古文，或者叫孩子就学不到蒙古文了的话，这种情况如果持续 5-10 年或更长的话，具有限制一代人蒙古文学习使用的可能性、降低一代人学习使用蒙古文的热情的负面效应。导致蒙古文使用用户急剧减少，乃至逐渐消失的危险性。

三、 从编码原则考虑蒙古文编码应该采用纯粹“形码方案”

3. 历史上蒙古文印刷方式全部都是采用字形作为最小单位

蒙古文的印刷行业经历了木刻版、铅字版、打字机、电子打字机、计算机打印、桌面排版印刷系统等历史时期。在这些历史时期，蒙古文的最小印刷单位都采用了不同的字形。我们在此看看电子化之前的蒙古文印刷方式采用的字形。

●铅字版

目前还没有找到蒙古文“铅字版”排版的实物图片。下图 1 是铅字版排版的蒙古文图书中的一段文字。

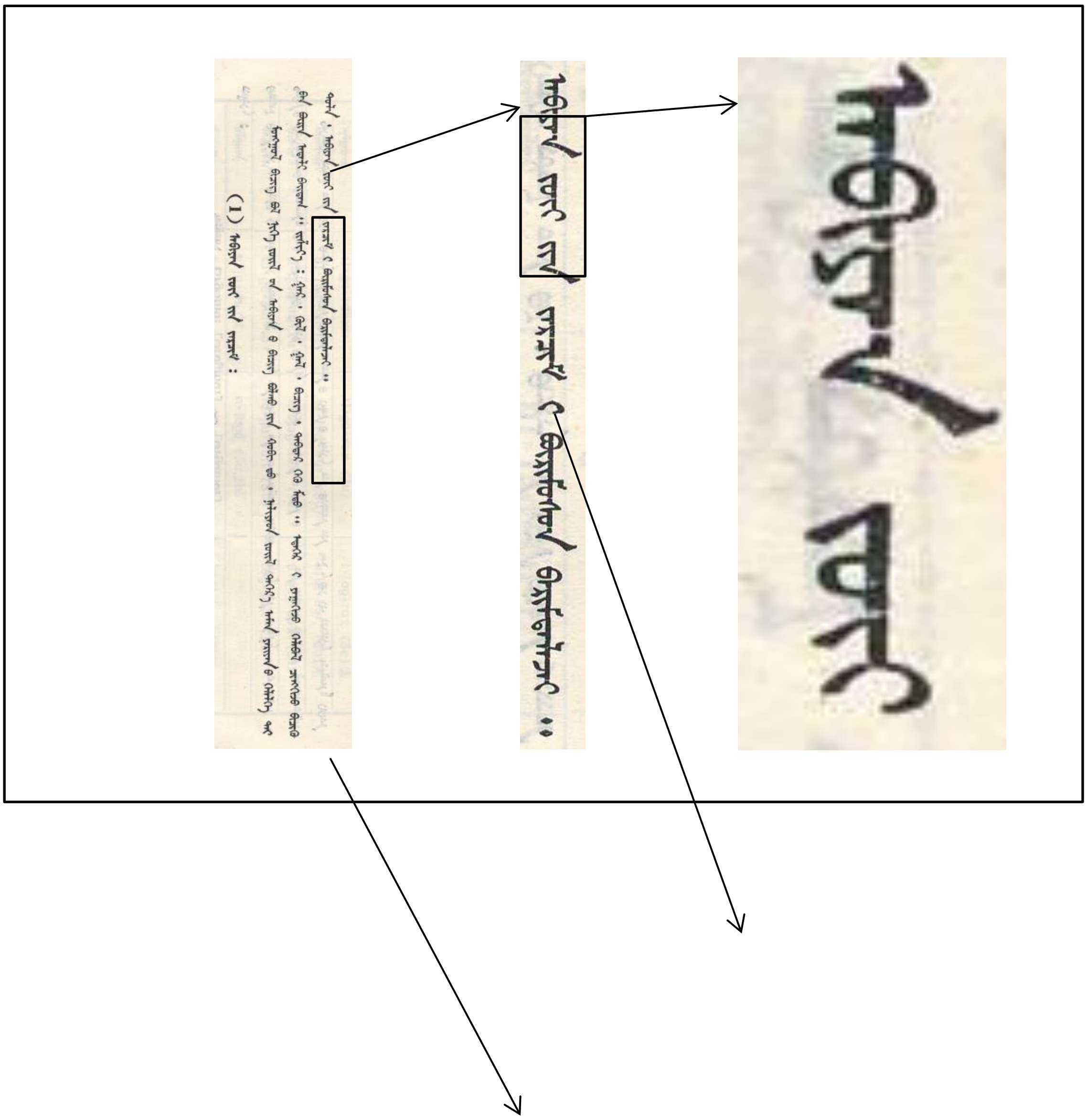


图 1　铅字版蒙古文样张

从图 1 的铅字版蒙古文样张看到，每一个铅字字模的痕迹。例如上图的最后两个单词看到的字模有： 是单独的一个字模。能看出铅字版为了捡字排版的效率考虑，主要按蒙古文音节字形作为最小单位字模。

●打字机

下图 2 是我们找到的一部蒙古文打字机的图片。其键盘布局为如图 3.该打字机实物现在在内蒙古德力海公司办公室存放。

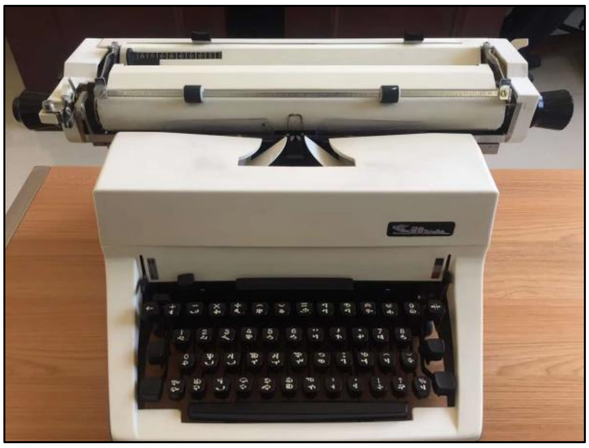


图 2 蒙古文打字机

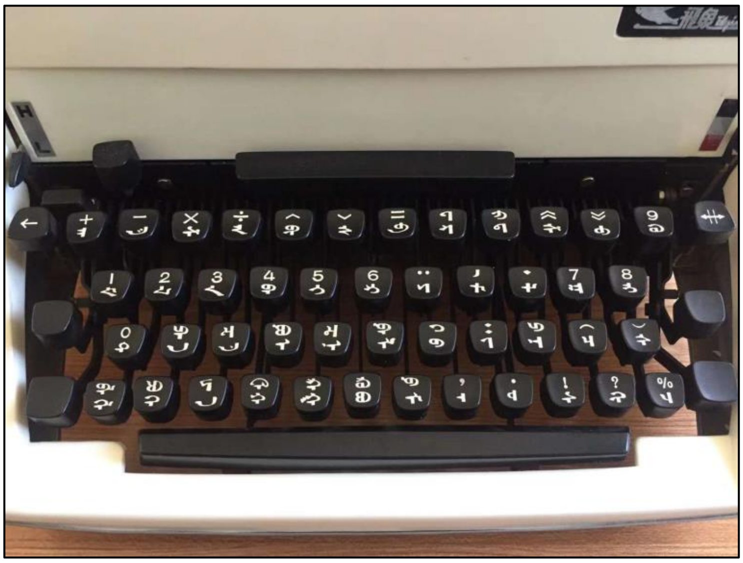


图 3 蒙古文打字机键盘布局

从图 3 看出，因为打字机可安排键盘个数有限，选择了蒙古文的很多字形部件以及字形无法拆分的合体字作为最小字形。

4. 早期计算机系统的蒙古文实现也都是形码

自从上世纪 80 年代开始研究蒙古文在计算机上的录入、显示与打印，出现了很多种蒙古文应用系统。这些系统无一例外采用了蒙古文的字形作为编码单位，也就是说都是“形码”编码。例如：如下所列系统都是蒙古文“形码”编码

✧ 早期 MWDOS 系统（如图 4 所示）

✧ 华光排版系统

✧ 方正排版系统

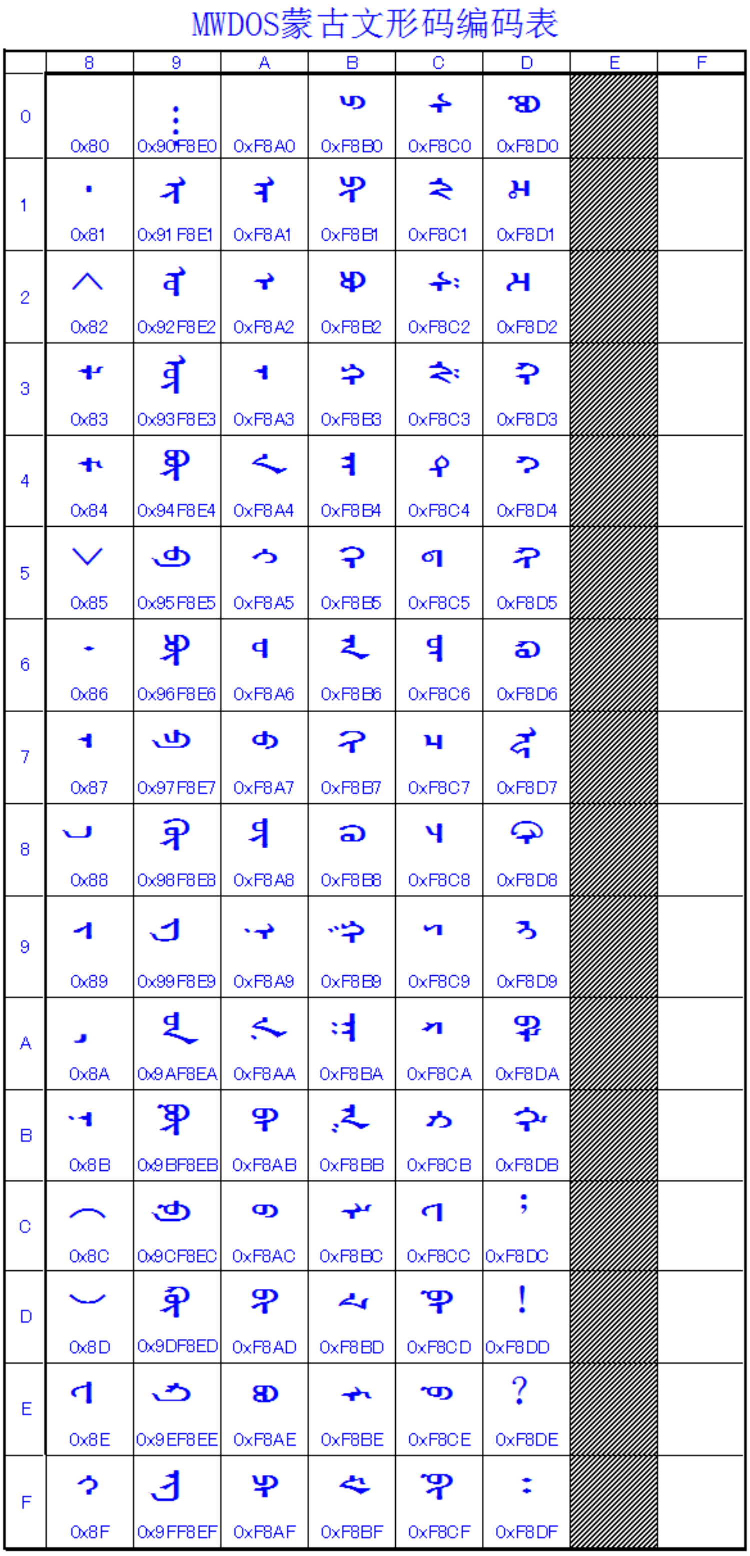
✧ 蒙科立 2002 系统

✧ 赛音编码系统等等

并且这些系统多数针对所有的字符变形与强行合体字进行了按“字形”的不同唯一编码。唯独蒙科立 2002 系统采用了完全按字符变形编码，合体字也是字形的两个字形相连接的方法实现的“形码”编码系统。

重要的一点是：这些系统都把 ᠠ ᠡ ᠢ ᠣ ᠥ 等元音字形单独编码。是因为我们在将近几百年的历史当中，除了蒙古文打字机的键盘个数限制的情况以外，都把它当作一个独立的最小字形单位。如果我们在编码方案中看不到这些最小字形单位的元音，从文化传承的意义考虑感情上有点接受不了。

如果，UNICODE 编码为了码位数量控制，一定要用蒙古文的字形部件编码，那也必须是纯粹的蒙古文“部件”形码，一旦引进蒙古文“部件”的上中下变化，那就是蒙古文固有规律的篡改。蒙古人很难按原有的语言知识正确使用该编码。这个也是 2017 年 9 月份 WG2 蒙古文专题会议上得到所有蒙古人反对的理由之一。包括我在内，都很难接受引进蒙古文“部件”的上中下变化方案。



MWDOS蒙古文形码编码表

	8	9	A	B	C	D	E	F
0	0x80	⋮ 0x90F8E0	0xF8A0	0xF8B0	0xF8C0	0xF8D0	▨	
1	· 0x81	0x91 F8E1	0xF8A1	0xF8B1	0xF8C1	0xF8D1	▨	
2	∧ 0x82	0x92F8E2	0xF8A2	0xF8B2	0xF8C2	0xF8D2	▨	
3	0x83	0x93F8E3	0xF8A3	0xF8B3	0xF8C3	0xF8D3	▨	
4	0x84	0x94F8E4	0xF8A4	0xF8B4	0xF8C4	0xF8D4	▨	
5	∨ 0x85	0x95F8E5	0xF8A5	0xF8B5	0xF8C5	0xF8D5	▨	
6	0x86	0x96F8E6	0xF8A6	0xF8B6	0xF8C6	0xF8D6	▨	
7	0x87	0x97F8E7	0xF8A7	0xF8B7	0xF8C7	0xF8D7	▨	
8	0x88	0x98F8E8	0xF8A8	0xF8B8	0xF8C8	0xF8D8	▨	
9	0x89	0x99F8E9	0xF8A9	0xF8B9	0xF8C9	0xF8D9	▨	
A	0x8A	0x9AF8EA	0xF8AA	0xF8BA	0xF8CA	0xF8DA	▨	
B	0x8B	0x9BF8EB	0xF8AB	0xF8BB	0xF8CB	0xF8DB	▨	
C	0x8C	0x9CF8EC	0xF8AC	0xF8BC	0xF8CC	; 0xF8DC	▨	
D	0x8D	0x9DF8ED	0xF8AD	0xF8BD	0xF8CD	! 0xF8DD	▨	
E	0x8E	0x9EF8EE	0xF8AE	0xF8BE	0xF8CE	? 0xF8DE	▨	
F	0x8F	0x9FF8EF	0xF8AF	0xF8BF	0xF8CF	: 0xF8DF	▨	

图 4 早期 MWDOS 的编码表

四、 “阿拉伯连写模型”下的蒙古文形码方案的存在问题

1. 蒙古文独有的的字形变化特性

蒙古文的连写特性，虽然是从阿拉伯文的连写特性流传过来的，但是与阿拉伯文的“只有唯一的上中下变形字形特性”无法满足“蒙古文连写字形变换特性”的要求。

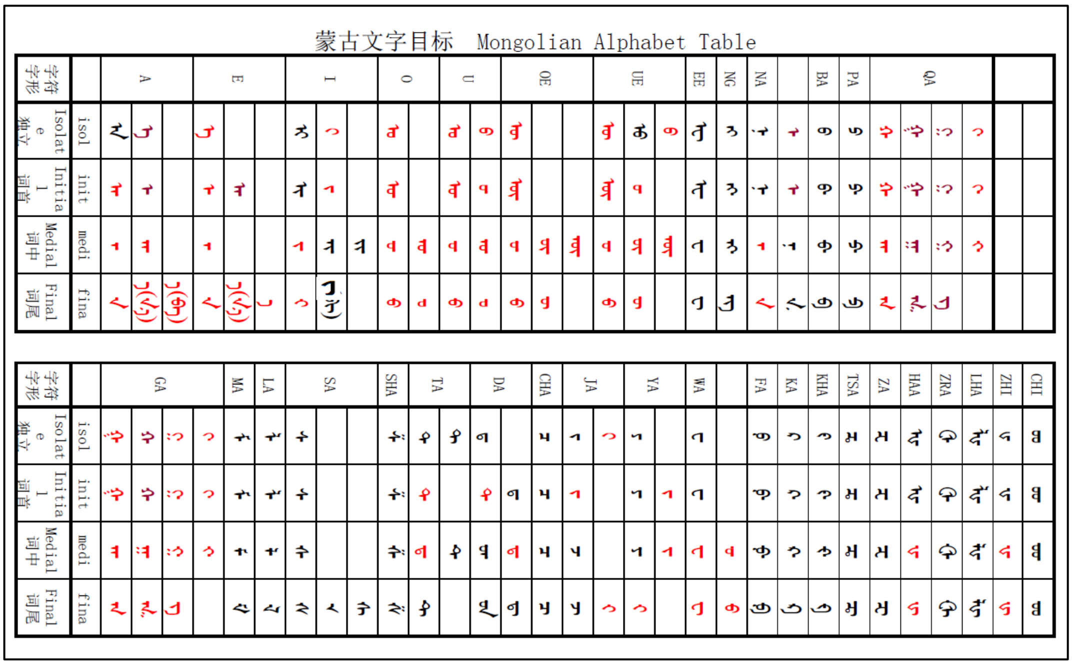

蒙古文字目标 Mongolian Alphabet Table
字符 字形
Isolate 独立
Initial 词首
Medial 词中
Final 词尾
isol
init
medi
fina
A
E
I
O
U
OE
UE
EE
NG
NA
BA
PA
QA
GA
MA
LA
SA
SHA
TA
DA
CHA
JA
YA
WA
FA
KA
KHA
TSA
ZA
HAA
ZRA
LHA
ZHI
CHI

图 5 蒙古文字母表

如上图 5 所示，蒙古文字母表中的红色字形都是与别的字母的变形具有相同字形的变形显现字形(其中暗红色字形是基本字母表中不存在的语言专家的特殊定义字形)。

例如：以下字形都是不同字母共享的字形。

字形	字母变形代号
	A-isol,E-isol;
	A- init,E-init;
	E-init,A-Init;
	A-medi,E-medi,NA-medi;
	A-medi,QA-medi,GA-medi;
	A-fina,E-fina,NA-fina;
	A-fina,E-fina;
	A-fina,E-fina;
	I-medi,YA-medi;
	I-fina,JA-fina,YA-fina;
	I-isol,JA-isol;

	I-init,YA-init;
	O-isol,U-isol;
	O-init,U-init;
	U-init,UE-init
	O-medi,U-medi,OE-medi,UE-medi,WA-medi;
	O-medi,Umedi;
	O-fina,U-fina,OE-fina,UE-fina,WA-fina;
	O-fina,U-fina;
	U-isol,UE-isol;
	OE-isol,UE-isol;
	OE-init,UE-init;
	OE-medi,UE-medi;
	OE-medi,UE-medi;
	OE-fina,UE-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;
	QA-fina,GA-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;
	QA-medi,GA-medi;
	QA-fina,GA-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;
	QA-medi,GA-medi;
	QA-fina,GA-fina;
	QA-isol,GA-isol;
	QA-init,GA-init;
	QA-medi,GA-medi;
	TA-init,DA-init;
	TA-medi,DA-medi;
	EE-medi,WA-medi;
	EE-fina,WA-fina;
	HAA-medi,ZHI-medi;
	HAA-fina,ZHI-fina;

2. “阿拉伯连写模型”下的蒙古文编码引起的问题

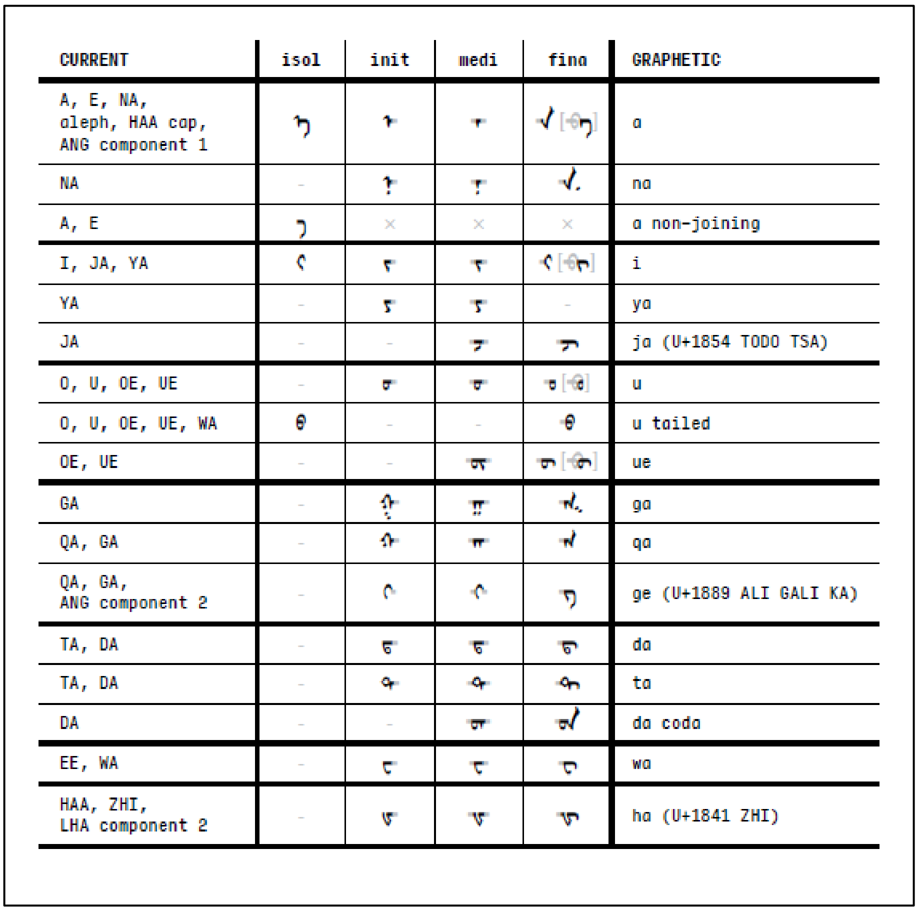
2.1 “音码方案”的“一形多码”问题

在蒙古文编码中利用“阿拉伯连写模型”，“不同字母”的“上中下字形”中存在很多“相同字形”，从而导致所见“内容”的编码多样性。这是现在的蒙古文编码普遍存在的问题。对于这个问题的改善方案暂且不在这里讨论，在别的文档中另行提出。

2.2 “形码方案”的字符串操作问题

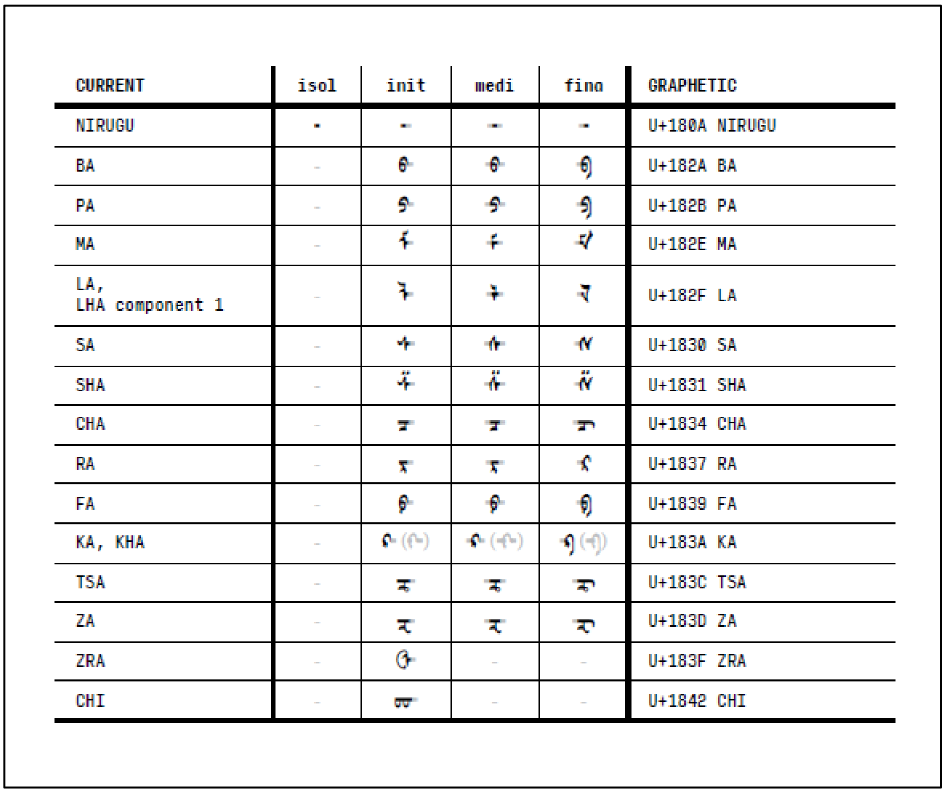
如果我们在形码方案中采用“阿拉伯连写模型”，那就是剥离字形与字母的关系，只留下一个字形是唯一个“代表字形”的上中下字形关系。并且只对该“代表字形”编码。一个“代表字形”的上中下字形也分别是唯一的。就像 WG2 N4889 那样。暂且不说，WG2 N4889 的“代表字形”选择的合理不合理，先来看看这一编码的致命性问题，即字符串操作的问题。

为了易于参照，我在此复制了 N4889 文档中的编码表。如图 6 和图 7 所示。



CURRENT	isol	init	medi	fina	GRAPHETIC
A, E, NA, aleph, HAA cap, ANG component 1					a
NA	-				na
A, E		×	×	×	a non-joining
I, JA, YA					i
YA	-			-	ya
JA	-	-			ja (U+1854 TODO TSA)
O, U, OE, UE	-				u
O, U, OE, UE, WA		-	-		u tailed
OE, UE	-	-			ue
GA	-				ga
QA, GA	-				qa
QA, GA, ANG component 2	-				ge (U+1889 ALI GALI KA)
TA, DA	-				da
TA, DA	-				ta
DA	-	-			da coda
EE, WA	-				wa
HAA, ZHI, LHA component 2	-				ha (U+1841 ZHI)

图 6 N4889 编码表前半部分



CURRENT	isol	init	medi	fina	GRAPHETIC
NIRUGU	᠊	᠊‍	‍᠊‍	‍᠊	U+180A NIRUGU
BA	-	ᠪ‍	‍ᠪ‍	‍ᠪ	U+182A BA
PA	-	ᠫ‍	‍ᠫ‍	‍ᠫ	U+182B PA
MA	-	ᠮ‍	‍ᠮ‍	‍ᠮ	U+182E MA
LA, LHA component 1	-	ᠯ‍	‍ᠯ‍	‍ᠯ	U+182F LA
SA	-	ᠰ‍	‍ᠰ‍	‍ᠰ	U+1830 SA
SHA	-	ᠱ‍	‍ᠱ‍	‍ᠱ	U+1831 SHA
CHA	-	ᠴ‍	‍ᠴ‍	‍ᠴ	U+1834 CHA
RA	-	ᠷ‍	‍ᠷ‍	‍ᠷ	U+1837 RA
FA	-	ᠹ‍	‍ᠹ‍	‍ᠹ	U+1839 FA
KA, KHA	-	ᠺ‍ (ᠻ‍)	‍ᠺ‍ (‍ᠻ‍)	‍ᠺ (‍ᠻ)	U+183A KA
TSA	-	ᠼ‍	‍ᠼ‍	‍ᠼ	U+183C TSA
ZA	-	ᠽ‍	‍ᠽ‍	‍ᠽ	U+183D ZA
ZRA	-	ᠿ‍	-	-	U+183F ZRA
CHI	-	ᡂ‍	-	-	U+1842 CHI

图 7 N4889 编码表后半部分

大家都是编码专家，我就不解释什么叫“字符串操作”了。我想做的是 N4889 能否保证以下两个字符串操作让用户买一。

(1) 两个字符串连接之后能否得到正确显示的蒙古文？在按第一个字符串的长度在切分之后能否得到与连接之前字符串同样的字符串？

我们先看看其他语种对这个问题的答案是什么？

英语：rain + bow => rainbow => split(4)= rain + bow；

中文：人民+公社 => 人民公社 => split (2) = 人民 + 公社

日文：お困り+でした => お困りでした => split (3) = お困り+でした

下面看看 N4889 编码的蒙古文得到什么样的结果？

(a) ᠠᠠ + ᠠᠢᠳᠠᠰ = ? 应该得到什么样的字符串？

（蒙古人会认为应该得到 ᠠᠠᠠᠢᠳᠠᠰ ）

但是，从 N4889 编码看应该得到什么？

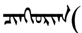
ᠠᠠ (a+a) + ᠠᠢᠳᠠᠰ (a+i+da+a+sa) = ᠠᠠᠠᠢᠳᠠᠰ(a+a+a+i+da+a+sa)

与预想的不一样！但有点靠边。

(b) + = ? 应该得到什么样的字符串？

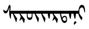
（蒙古人会认为应该得到 ）

但是，从 N4889 编码看应该得到什么？

(ca+i+a+ge) + (i+u+ra+i+qa) = (ca+i+a+ge+i+u+ra+i+qa)

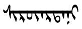
出乎意料，不可想象！！！

(c) + = ? 应该得到什么样的字符串？

（蒙古人会认为应该得到 ）

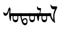
但是，从 N4889 编码看应该得到什么？

(sa+i+ra+u+i) + (a+ra+da+a+na+i) =

 (sa+i+ra+u+i+ a+ra+da+a+na+i)

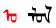
与我们的预想差距太大，导致蒙古人很难接受啊！

(c) + = ? 应该得到什么样的字符串？

（蒙古人会认为应该得到 ）

但是，从 N4889 编码看应该得到什么？

(sa+u+da+u-tailed) + (la+u+la) =

 (sa+u+da+u-tailed+la+u+la)，u-tailed 在词中不知怎么显示！

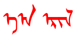
这个是我们不想看到字形，一般情况下我们不想看到，在一个蒙古文单词的“词中”出现“词尾形”！

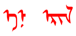
当然，上述字符串按原来子字符串的长度切分，肯定得到与原字符串一样的子字符串。

想解决这些连接后的单词与预想结果不一样的问题，引进组合后单词的智能化修正的话，麻烦会变得更大、更复杂。

虽然引进智能化修正的解决办法是不可行的，我也按作者介绍的利用智能编辑软件帮助纠正字符串操作（连接与切分）的假设看看一个例子的情况。

假设，原来有一个蒙古文字符串是 把它切分，智能化处理，再连接就不知道该怎么智能化处理连接后的字符串了。

(a+a)+ (a+ra+i+l) =>  (a+a+a+ra+i+l) (anril)？

(a+a)+ (a+ra+i+l) =>  (a+na+a+ra+i+l) (eneril)？

所以字符串操作与字符串编辑过程中不可能采用智能校对与智能编辑帮助系统保证“字符串正确性”。对于 N4889 编码可取的智能帮助是只能在录入阶段有

用，而在编辑阶段没有任何帮助，有可能帮倒忙。

(2) 一个字符串切分之后，子字符串的显示结果会是什么样的呢？一个长字符串按指定长度切分之后，再连接是否保证得到与原先字符串完全一样的字符串？

我们先看看其他语种对这个问题的答案是什么？

英语：speedup => split(5)=> speed + up => speedup；

中文：联合作业 => split (2) => 联合+作业 =>联合作业；

日文：迷惑をかける => split (3) =迷惑を+かける =>迷惑をかける；

下面看看 N4889 编码的蒙古文得到什么样的结果？

(a) => split(5)=> ？ 蒙古人会预测为(, +)

但是实际得到的是 + ，还是与预测有距离！

=> => split(6)=> ？ 蒙古人会预测为(+)

但是实际得到的是 + ，不该断字的地方可断字了！

切分的子字符串再连接，当然准确无误地得到原字符串 => ，当然这是在不采用智能手段纠正子字符串的情况下的结果。

(b) => split(3)=> ？ 蒙古人会预测为(+)

但 (c) 我想输入蒙古文的复合词 利用现在熟悉的音码输入(agasi jang)，但是由于中间空格没有能够按到变成了(agasijang)，根据输入音码到 N4889 的编码的智能转换，应该为 最后发现空格错了，光标退回到 的前边加一个空格会变成什么？

=> split(6)=> ？ 蒙古人会预测为( )

但是实际得到的是 ，用户不知道该怎样把这个 变成词首的 ？

当知道必需把 重新输入，让输入法帮助转换的时候会说“不！”的。

2.3　N4889 的“形码方案”也没有完全解决“一形多码”的问题

(a) **(词中 qa)　和** **(词中 a+a)　一样**

(　(c) **(词尾 ue)　和** **(词尾 u+a)　一样**
b　(d) **(词首 CHI)　和** **(词首 u+u)　一样**
)

2.4　“形码方案”的最佳选项是采用“纯粹形码方案”

-从以上的讨论看，采用“阿拉伯连写模型”的“音码方案”和“形码方案”都有其缺点。这是因为蒙古文的固有特性所决定的，不可忽视的特点。

如果想用“形码编码”，并且想完全解决这些这些问题，那应该采用“纯粹的形码方案”是最佳选项。并且，在内蒙古已经有多年的应用经验。可以充分分析以前应用中的优缺点，设计一个可行的并且容易被蒙古族民众接受的方案才是最理想化的选择。

五、 对 WG2 N4889 方案存在问题的质疑点

1. 蒙古文的最常用的头三个字符的 A、E、I 没有被独立编码

首先从所有蒙古民族对自己文字的认识与理解角度看，这个方案是个对蒙古文字进行部件化拆分之后的编码方案。但是，蒙古民族的文字历史上有人撰写过这样解释的文章，但不是在教学过程广泛采用的办法。在我们的从小学到高中的教学中从来没有提到过这样的解释。因此，对于这样分解，大部分蒙古族民众是陌生的概念。因此，这个编码方案，可能导致另一种文字改革的风险。

我们从小学习得到的语言文字学得的知识出发，并根据近 40 年的信息化研究经验认真考虑的话，我们蒙古文的编码表里，至少应该把蒙古文的头三个字母ᠠᠡᠢ或者ᠠ‍ᠡ‍ᠢ‍(A、E、I)单独编码到 UNICODE 的码位上。这是任何一个文字都要求的，最基本的要求。

虽然，N4889 能够把这几个元音都能显示出来，但是只有一个 E 和半个 I，没有 A 的方案是很难接受的。一个语言如果在长久的历史潮流中，A 和 E 不分了的话，是一种最大的危机。日语就是典型的例子。因为，日语中没有 E 这个元音，所有日本人，如果没有在国外受过其他语种的教育，他们是分不清，或绝对分不了 A、E 的区别。都发音为 A（あ）、导致学习外语时的最大的障碍。

蒙古语是世界上为数不多的多元音语言，传统元音 7 个之上，外借元音 ᠧ（EE）总共有 8 个元音。这还没有包含有些专家所说的半元音ᠶ᠂ᠸ（YA、WA）。我们很担心，N4889 方案会把我们蒙古文的这一优点埋在编码的底层，导致将来有一天会把蒙古文的元音丢失在历史过程中。因为，只有记录发音的符号才能把该发音传承下去。我虽然不懂日语的发展历史，但总认为日语的发音中在历史上肯定也有过发 E 的音的时期或人，只因为文字中没有记录这个音的符号，长期以来的教学过程中，日本人就不知道发 E 这个音，久而久之在日本 E 这个音就被丢掉了。他们会很自然地把 Earth 发音为(アース)造成英语世界的人听不懂。（很抱歉，这里不是在说日语的不好，而是说明重视记录元音符号的存在的意义。比起我们的蒙古语，在其他方面日语比我们先进很多，我们都是在天天向他们学习先进的理论概念以及技术）。

2. 对于不存在的独立形以及上中下字形的位置都应该给出指定的字形。

在 N4889 中，对于不存在的独立形以及上中下字形的位置都没有给出明确的字

形。从“阿拉伯连写模型”的定义看，这样的“无效字符”的定义办法不可取。

首先、任何一个字符（Unicode 的码位代码）都有可能在独立位置，或一个单词的词首、词中、词尾出现的可能性。那么、这些“这些无效”定义字符出现时，如果显示的是统一的一个字形或留空（像在呼和浩特会议上的表演中用十字架代替了这个“无效字符”），那么谁也判断不了这个“无效字符”的后面隐藏的代码了。这个不就是我们通常说的“安全隐患”吗？

3. 文字输入与文本编辑问题

（a）文字输入问题

首先，在呼和浩特的 WG2 蒙古文会议上介绍 N4889 编码方案时提到过在蒙古文输入过程中不改变用户体验。但是，根据我们的详细琢磨，不改变用户体验的说法太牵强了。在 N4889 中，完全放弃了 MVS、NNBSP、FVS1-3 的使用。那么我想问，不使用 MVS 等辅助键的指定的情况下如何实现分写形式的 A 呢？即利用与现行输入法相同用户体验的情况下，请给予解释下面两个单词的输入方法。

ᠰᠠᠷᠠ (sara？)　　　　　　保存为 ᠰ (sa) + ᠠ (a) + ᠷ (ra) + ᠠ (a)

ᠰ ᠠ ᠷ ᠠ (sara?sar<MVS>a?)

（b）文本编辑问题

在前面的“字符串操作使得问题”中也已经谈到了 N4889 编码方案在“文本编辑”时存在问题。也就是“用户在编辑过程中所得到的结果与自己所掌握的语言文字学知识为前提的预想内容完全不相同”的用户体验。因此我们怎么能说『用户完全不用意识“形码方案”后面的实际编码，而只按自己原有的知识就可以顺利使用』的说法能成立呢？

另外、用户对 N4889 编码方案的完全理解之前，“文本编辑”的效率也不可能与原来一样。完全不可能按原来的知识去准确修改一个单词中的错误字符。

与 N4889 的作者辩论的时候，“提到过目前系统都提供获取光标下单词提交给输入法的功能”的辩解。我知道 Windows 的某些应用有这样的功能。但其他系统有没有这样的功能仍然是未知数。我们不能再一次等待所有系统提供类似的技术和功能，才能达到高效率使用 N4889 蒙古文“形码方案”编码标准！

蒙古文 N4889 编码方案在“文本编辑”时出现不可预见的麻烦与问题是影响用户体验效果、降低工作效率的不可忽视的问题。

4. “字符界限”不明确的问题

N4889 编码方案是采用蒙古文字形的部件作为最小单位（码位）的编码方案。为此很多蒙古文的字符（字母）需要用两个或两个以上的码位组合表示。例如：ᠲ ᠪ ᠫ ᠬ ᠲ ᠲ ᠲ ᠳ ᠳ ᠳ ...... 等都是用两个或两个以上字符组成。如果按码位的单位去分解和切分的话变成可行了。但是这些字符在蒙古文中是不可分割的完整独立字形。

不能从中间被分割或切分的。但是，N4889 方案没有准确定义“字符界限”的确定办法。

请各位专家能否介绍一下 UNICODE 中其他文字如何确定“字符界限”的？蒙古文的编码 N4889 需不需要确定“字符界限”的准确定义为好？

5. 字符串操作问题

字符串操作的问题前面已经讲过了。这里就不再重复了。

6. 【a non-joining】的属性定义的细节问题

我们发现 N4889 的码位【a non-joining】的 init、medi、fina 都是【X】,能否解释该字符（码位）的 UNICODE 字符属性？

如何判断这个字符前面的字符应该是“词尾形”而不是“词中形”？

是字体渲染引擎在判断吗？UNICODE 的类似这样的字符属性定义文件在哪里能找到？

是不是可以理解唯独这个字符是不采用“阿拉伯连写模型”的“纯粹的形码”？

N4889 中标记【-】的和【X】的区别是什么？

7. 与现有标准的共同存在问题

N4889 编码方案没有给出与“现行蒙古文编码方案”共同存在的可行性研究。如果不与“现行蒙古文编码方案”共存并能够并行使用的话，那这个新方案不会被用户群体接受的。

8. 与现有标准的双向转换问题

N4889 编码方案也没有给出与现行“蒙古文编码方案”双向转换的可行性研究。如果不支持与“现行蒙古文编码方案”相互转换的话，那这个新方案的推广也会受到很大阻力的。

五、 蒙古文“形码方案”提案的理想化步骤与目标

如果决议要推行“形码方案”的蒙古文标准的话，最理想化的步骤与目标应该是：

首先，对目前现行的蒙古文 “音码方案”的编码标准的稳定以及统一的早日实现。

其次，设计理想的蒙古文“形码方案”编码标准达到新方案标准与目前现行的“音码方案”编码标准能够共存并能够互转。

最后，逐渐启用蒙古文“形码方案”编码标准，废弃“音码方案”编码标准。

将来，蒙古文“音码方案”编码标准部分可以成为语言学研究等方向的辅助编码标准方案。

最终，我们也都希望我们的蒙古文尽早有一个完整的、稳定的、统一的、理想的“编码标准”。其前提是对目前管饭使用的编码标准起冲击作用。精明设计，稳步实施，并行运行，逐步切换才能够被用户接受。

注：同意此文观点者有：斯琴毕力格（siqin@almas.co.jp）、包海山（baohaishan@delehi.com）、布日古都（burgood@vip.sina.com）等