Technical Reports

L2/18-298

Draft Unicode Technical Report #53

UNICODE ARABIC MARK RENDERING

Authors	Roozbeh Pournader (roozbeh@unicode.org), Bob Hallissy (bob_hallissy@sil.org), Lorna Evans (lorna_evans@sil.org)
Date	2018-07-16
This Version	http://www.unicode.org/reports/tr53/tr53-3.html
Previous Version	http://www.unicode.org/reports/tr53/tr53-2.html
Latest Version	http://www.unicode.org/reports/tr53/
Latest Proposed Update	http://www.unicode.org/reports/tr53/proposed.html
Revision	3

Summary

This technical report specifies an algorithm that can be utilized during rendering for determining correct display of Arabic combining mark sequences.

This UTR makes no change to Unicode normalization forms, and does not propose a new normalization form. Instead, this is similar to the processing used in [MicrosoftUSE]: a transient process which is used to reorder text for display in an internal rendering pipeline. This reordering is not intended for modifying original text, nor for open interchange.

Status

*This is a **draft** document which may be updated, replaced, or superseded by other documents at any time. Publication does not imply endorsement by the Unicode Consortium. This is not a stable document; it is inappropriate to cite this document as other than a work in progress.*

> ***A Unicode Technical Report (UTR)** contains informative material. Conformance to the Unicode Standard does not imply conformance to any UTR. Other specifications, however, are free to make normative references to a UTR.*

Please submit corrigenda and other comments with the online reporting form [Feedback]. Related information that is useful in understanding this document is found in the References. For the latest version of the Unicode Standard see [Unicode]. For a list of current Unicode Technical Reports see [Reports]. For more information about versions of the Unicode Standard, see [Versions].

Contents

---

1 Overview

The combining classes of Arabic combining characters in Unicode are different than combining classes in most other scripts. They are a mixture of special classes for specific marks plus two more generalized classes for all the other marks. This has resulted in inconsistent and/or incorrect rendering for sequences with multiple combining marks since Unicode 2.0.

The Arabic Mark Transient Reordering Algorithm (AMTRA) described herein is the recommended solution to achieving correct and consistent rendering of Arabic combining mark sequences. This algorithm provides results that match user expectations and assures that canonically equivalent sequences are rendered identically, independent of the order of the combining marks.

2 Background

Rules and recommendations for the correct display of combining marks are discussed in a number of places in the Unicode Standard, including sections 5.13, 7.9, and 9.2 [Unicode]. Some general principles include:

- Canonically equivalent sequences should display the same.
- Combining marks from the same combining class are normally displayed using the *inside-out* rule, i.e., from the base outward.
- Combining marks from different combining classes (other than ccc=0) may be re-ordered with respect to each other if that helps to achieve the desired display.

In Unicode, the Arabic script combining marks include eleven different non-zero canonical combining class values, as shown in Table 1. When a combining character sequence includes marks from more than one of these classes, the rendering system has to determine a display order in which to position these marks on the base character.

While it might be tempting to just use NFC or NFD, neither of these normalization forms will yield what Arabic readers expect. For one example that will be easily understood by all readers of Arabic script, given a combining character sequence including a shadda (ccc=33) and damma (ccc=31), NFC and NFD will move the *damma* before the *shadda* – at which point the default inside-out rendering rule would place the *shadda* above the *damma*, which is incorrect.

Some cases are obvious to readers of languages written with Arabic script, and thus will likely get the same display from various rendering implementations. However, many of the combining marks, especially those with ccc=220 and ccc=230, are not commonly understood. Different rendering implementations have made different decisions regarding display order, resulting in inconsistent behavior between one system and another.

AMTRA defines a method to reorder Arabic combining marks in order to accomplish the following goals:

- The inside-out rendering rule will display combining marks in the expected visual order.

- Ensure identical display of canonically equivalent sequences.
- Provide a mechanism for overriding the display order in exceptional cases.

Table 1: Canonical combining class values for marks used in Arabic script

Canonical Combining Class (ccc) value	Combining marks in this class
0	Combining grapheme joiner
27	fathatan, open fathatan
28	dammatan, open dammatan
29	kasratan, open kasratan
30	fatha, small fatha
31	damma, small damma
32	kasra, small kasra
33	shadda
34	sukun
35	Superscript alef
220	All other below combining marks
230	All other above combining marks

3 Description of the Algorithm

The algorithm starts by reordering combining marks according to one of the Unicode Normalization forms, and then makes adjustments by moving certain marks closer to the base.

3.1 Modifier Combining Marks (MCM)

For use by this algorithm, we define a group of combining marks called “Modifier Combining Marks” (MCM). MCM are combining characters that are normally used to modify the base character before them, and should normally be rendered closer to the base character than *tashkil* (supplementary diacritics, including vowels). The MCM characters are not formally classified as *ijam* (consonant pointing/nukta, etc) in the Unicode Standard, but they are usually perceived by users as *ijam*.

The complete list of MCM characters is:

U+0654 ARABIC HAMZA ABOVE
U+0655 ARABIC HAMZA BELOW
U+0658 ARABIC MARK NOON GHUNNA
U+06DC ARABIC SMALL HIGH SEEN
U+06E3 ARABIC SMALL LOW SEEN
U+06E7 ARABIC SMALL HIGH YEH
U+06E8 ARABIC SMALL HIGH NOON
U+08F3 ARABIC SMALL HIGH WAW
U+08D3 ARABIC SMALL LOW WAW

The set of MCM characters is stable. Adding an existing Unicode character to the list of MCM could change the rendering of data that assumes the implementation of AMTRA. Additional characters may be added to the MCM at the time they are encoded (see section 5.4 Rationale for exclusion of some marks).

3.2 Specification of AMTRA

In the following specification, parenthetical definitions, for example *(D56)*, refer to definitions in the Unicode core specification.

Input: A Combining Character Sequence (D56) containing one or more Arabic combining marks.

Output: A canonically equivalent Combining Character Sequence reordered for rendering using inside-out stacking.

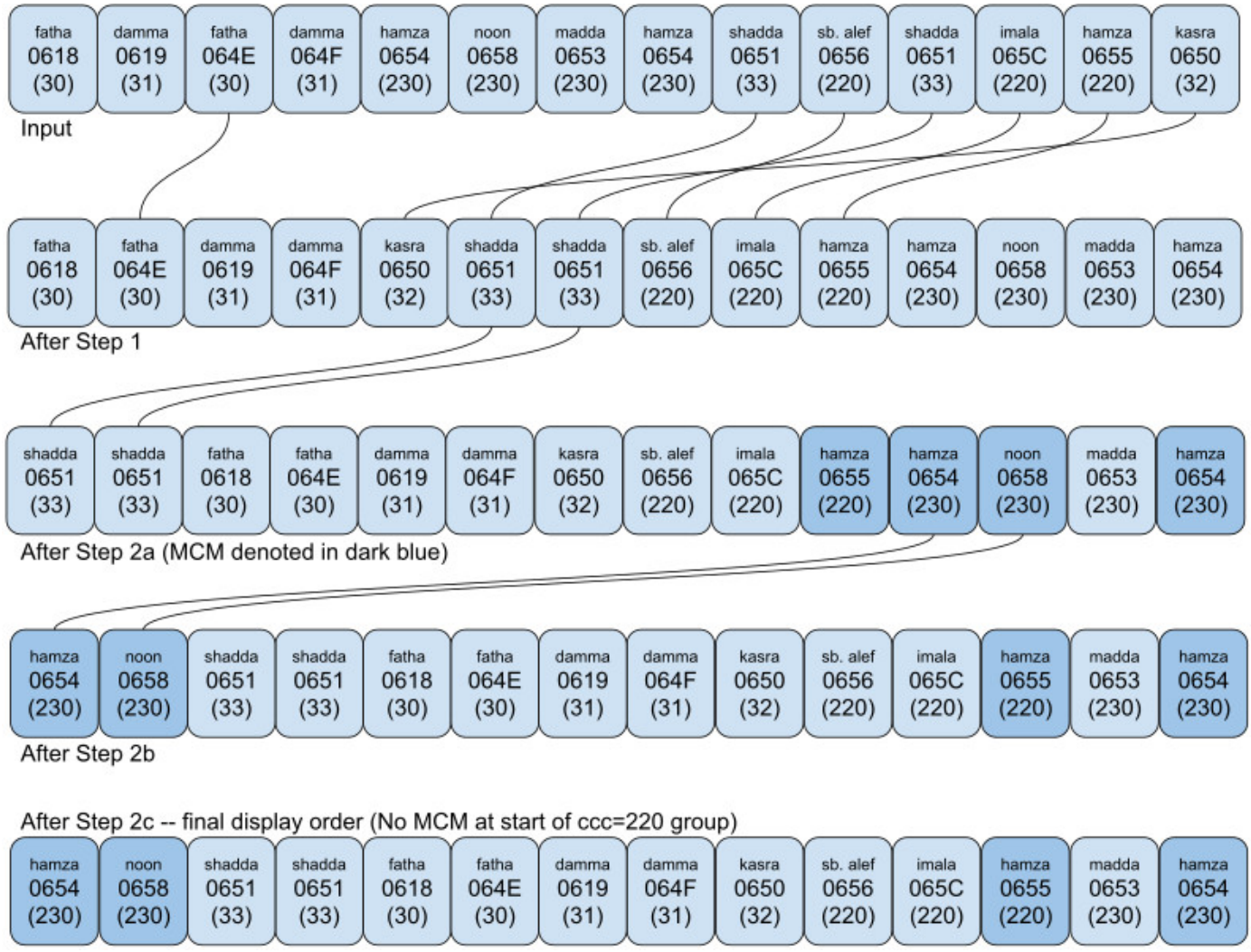
Steps:

1. Normalize the input to NFD
2. Within the result, for each maximal-length substring, S, of non-Starter (D107) characters, re-order as follows:
   a. Move any shadda characters (ccc=33) to the beginning of *S*.
   b. If a sequence of ccc=230 characters begins with any MCM characters, move the sequence of such MCM characters to the beginning of *S* (before any characters with ccc=33).
   c. If a sequence of ccc=220 characters begins with any MCM characters, move the sequence of such MCM characters to the beginning of *S* (before any MCM with ccc=230 or ccc=33).

4 Demonstrating AMTRA

4.1 Artificial Test Case

The following figure demonstrates the algorithm using an artificial sequence of characters:



4.2 Override Mechanism for Exceptions

The default display order implemented by the AMTRA will be correct for most uses. However in situations where a different mark order is desired, U+034F COMBINING GRAPHEME JOINER (CGJ) can be used to achieve the desired display order. The following sections give examples of the use of CGJ.

4.3 Examples

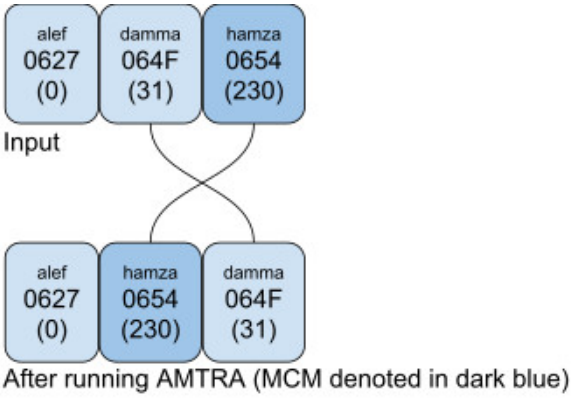
The following examples demonstrate why each of the respective characters is included in the MCM.

U+0654 ARABIC HAMZA ABOVE and U+0655 ARABIC HAMZA BELOW

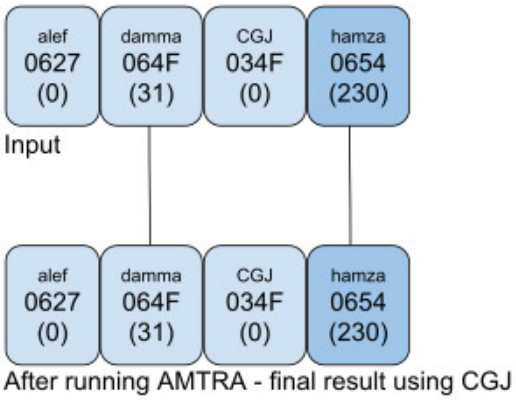
The use of combining hamza above and below is discussed in [Unicode, Chapter 9].

Example 1 [Quran1, page 9, end of line 5]

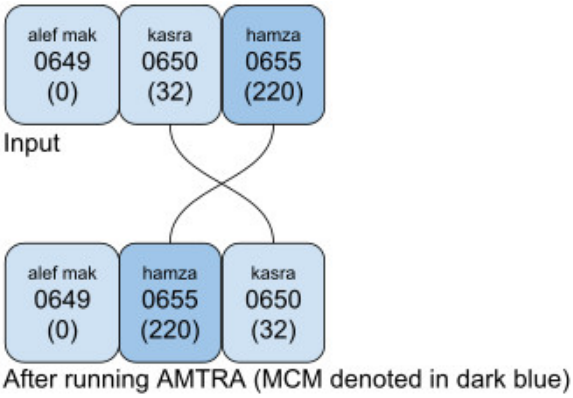
In Example 1, AMTRA puts a *damma* over a *hamza above*:



If an orthography needs to put the *hamza above* over the *damma*, the text should be encoded as <damma, CGJ, hamza above>:

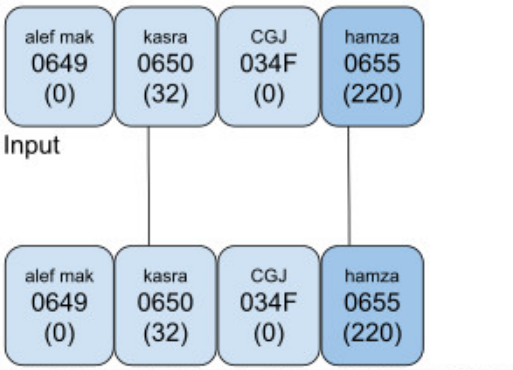


AMTRA puts the *kasra* below a *hamza below*:



After running AMTRA (MCM denoted in dark blue)

If an orthography needs to put the *hamza below* under the *kasra*, the text should be encoded as <kasra, CGJ, hamza below>:



After running AMTRA - final result using CGJ

U+0658 ARABIC MARK NOON GHUNNA

Regarding inclusion of this mark in the MCM, Kew says “The ARABIC NASALIZATION MARK is considered equivalent to a ‘nukta’, *as it is a modifier that binds tightly to the underlying letter;*” (italics added for emphasis) [Kew].The character is the character encoded as U+0658 ARABIC MARK NOON GHUNNA.

U+06DC ARABIC SMALL HIGH SEEN and U+06E3 ARABIC SMALL LOW SEEN

SMALL HIGH SEEN is included in MCM because most Quranic orthographies use the character as an MCM only. Orthographies that place the *small seen* differently will need to use a CGJ.

Example 2a [Al-Hilâlî]

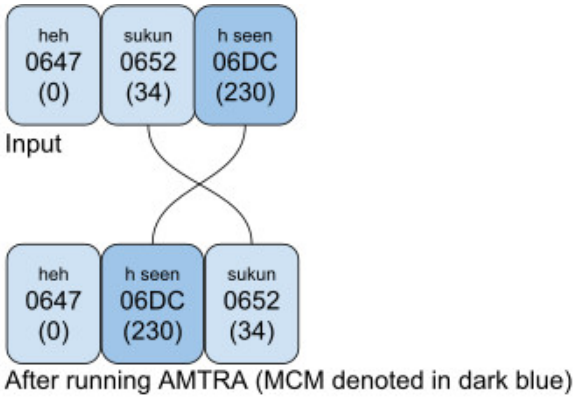
Example 2b [Al-Hilâlî]

In Example 2a, the *small high seen* is rendered below the *sukun*, while in Example 2b, it is rendered over it. The examples are indeed from the same document (Al-Hilâlî and Khân 1996), just two pages away. The *small high seen* has different roles: in Example 2a it is a hint that the base letter, *sad*, should be pronounced as if it was a *seen*; in Example 2b, it is a pause-related hint.

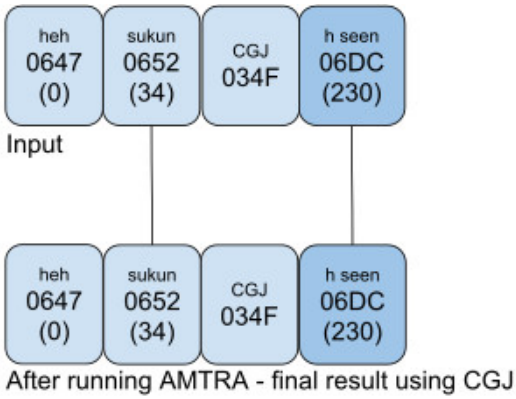
Example 2a (characters and properties):

Running AMTRA on this string does not result in any changes.

Example 2b (characters and properties):

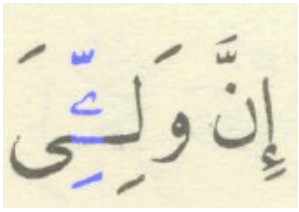


Running AMTRA on the string in Example 2b resulted in an undesired change. It puts a *sukun* over a *seen above*. If an orthography needs to put the *seen above* over the *sukun*, the text should be encoded as <sukun, CGJ, seen above>.

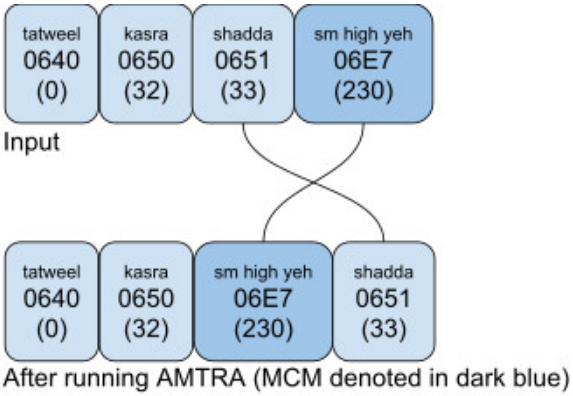


U+06E7 ARABIC SMALL HIGH YEH ~~and U+08F3 ARABIC SMALL HIGH WAW~~

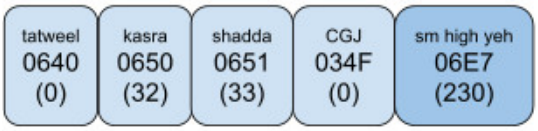
Example 3 [Milo, page 9, line 11]



In Example 3, AMTRA puts a *shadda* over a *small high yeh*.



If an orthography needs to put the *small high yeh* over the *shadda*, the text should be encoded as <shadda, CGJ, small high yeh>.



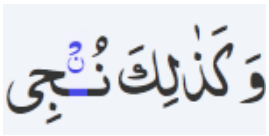
Running AMTRA on this string does not result in any changes.

U+08F3 ARABIC SMALL HIGH WAW and U+08D3 ARABIC SMALL LOW WAW

U+08F3 ARABIC SMALL HIGH WAW "is functionally similar to the already-encoded U+06E7 ARABIC SMALL HIGH YEH" and therefore *small high waw* is included in MCM [Pournader]. In available examples, *small high waw* and *small low waw* are functionally equivalent and, because they emphasize the vowel, are strongly bound to the body of the word. For these reasons they are both included in MCM.

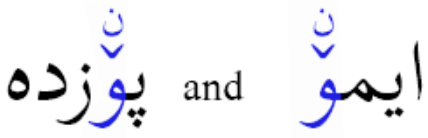
U+06E8 ARABIC SMALL HIGH NOON

Example 4a [Quran2]

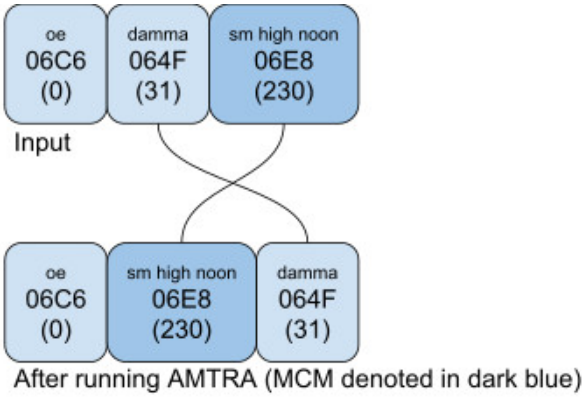


Example 4a has a *sukun* over a *small high noon*. AMTRA puts a *sukun* over a *small high noon*. If an orthography needs to put *small high noon* over *sukun*, the text should be encoded as <sukun, CGJ, small high noon>.

Example 4b



Example 4b shows a practical orthography that uses *small high noon* for nasalization. It is theoretically possible for a vowel to appear above the *small high noon* in this practical orthography. In such a case, AMTRA puts the vowel (in this case *damma*) above *small high noon*.



In order to force the *small high noon* above the vowel, use the CGJ (<oe, damma, CGJ, small high noon>).

5 Supplemental Information

5.1 Use of NFD and not NFC

NFD assures that sequences such as <superscript alef, madda> always result in the same ordering, independent of the base letter. If the algorithm were to use NFC instead, the sequence <alef, superscript alef, madda> would have resulted in a different order than <lam, superscript alef, madda>, because NFC composes <alef, madda> to <alef-with-madda-above>.

5.2 Shadda

The combining class for *shadda* (ccc=33) is higher than most vowels; however, it should be displayed closer to the base than the vowels.

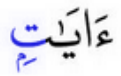
5.3 Kasra and kasra-like characters

AMTRA is able to handle the special ligation of *kasra* and *kasra-like* characters which are ligated with a *shadda* or *hamza* in some styles and appear just below them instead of below the base letter; they still logically follow the *shadda* or *hamza*.

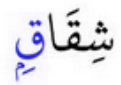
5.4 Rationale for exclusion of some marks

Meem above (ccc=230), *meem below* (ccc=220) and other similar characters are not included in the MCM because their behavior already meets normal expectations. Examples 5a-5c show that the *meem* is normally kept after *fatha*, *kasra* or *damma*, whereas including *meem above* and *meem below* in MCM would have the undesirable effect of moving them in front of *fatha*, *kasra* or *damma*.

Example 5a [Quran1, page 11]



Example 5b [Quran1, page 21]



Example 5c [Quran1, page 19]

Sukun alternate forms

There are three *sukun-like* marks encoded at U+06DF..U+06E1 that are used in some Quranic orthographies to denote different entities – they may not always represent a *sukun*. The canonical combining class of these marks is 230, so their ordering in the presence of other combining marks is not affected by AMTRA. However, since the combining class for the *sukun* is 34, these *sukun-like* marks will *not* be treated like a normal *sukun* in all cases. Users who create data using these alternate *sukun* characters will have more flexibility than when using the normal *sukun*. AMTRA does not make them equivalent to U+0652 ARABIC SUKUN, as that would make the algorithm unnecessarily complex and make the usage of CGJ more frequent.

Maddah

Neither U+0653 ARABIC MADDAH ABOVE (ccc=230) nor U+06E4 ARABIC SMALL HIGH MADDA (ccc=230) are MCM because they are normally displayed above vowel marks.

5.5 Dotted circles

Some rendering engines will insert a dotted circle for what it understands to be an invalid sequence. This is a problem in Arabic script because something that appears invalid may actually be valid text in some lesser known orthography of a minority language or in the Quran. For example, the Microsoft Windows text rendering engine, described in [Microsoft], inserts a dotted circle in combinations of certain Quranic marks that are known to appear with each other in the Quran.

Such spell-checking processes are best implemented at a higher level than a rendering engine. Also, a dotted circle insertion algorithm that displays all canonically equivalent sequences identically is hard to design and the result may be counter-intuitive for its users.

Implementations of the algorithm may be adapted to insert dotted circles by applying the algorithm first and then inserting the dotted circles.

5.6 Other uses for AMTRA

There is no intention or expectation that AMTRA would be applied to stored text. However, there may be situations unrelated to rendering where AMTRA may be useful, and this UTR does not prohibit such use.

As an example, when a text editor is processing a backspace key, a decision has to be made about what character(s) to remove from the text. For sequences involving combining marks, if the desire is to remove one mark at a time, users may have an expectation that the *outermost* marks should be removed first. For Arabic script the AMTRA could be used to identify outermost marks.

5.7 Best combining classes for yet-to-be-encoded combining marks in Arabic

When new combining marks are encoded, 220 should be used for below marks and 230 for above marks. In the special cases where an alternative version of the basic *tashkil* is encoded, the same combining class as the *tashkil* could be used, but extreme care should be taken.

References

[Al-Hilâlî] Muhammad Taqî-ud-Dîn Al-Hilâlî and Muhammad Muhsin Khân (translators) 1417 AH (=1996 CE). The Noble Qur'an: English Translation of the meanings and commentary. King Fahd Complex For The Printing of The Holy Qur'an. ISBN 9960-770-15-X.

[Feedback] Reporting Errors and Requesting Information Online http://www.unicode.org/reporting.html

[Kew] Kew, Jonathan, 2002. Bidi committee consensus on Arabic additions from L2/01-425 http://www.unicode.org/cgi-bin/GetMatchingDocs.pl?L2/02-061 (accessed 1 May 2017).

[Microsoft] Microsoft Typography 2014. Developing OpenType Fonts for Arabic Script. https://docs.microsoft.com/en-us/typography/script-development/arabic (accessed 16 Feb 2018).

[Microsoft USE] Microsoft Typography 2017. Creating and supporting OpenType fonts for the Universal Shaping Engine. https://docs.microsoft.com/en-us/typography/script-development/use (accessed 22 May 2018).

[Milo] Milo, Thomas. 2005. Annotations to the printing of the 1924 Azhar Qur'an (U+0670, U+06D6..U+06DB, U+06DD..U+06DF, U+06E0..U+06ED) http://www.unicode.org/cgi-bin/GetMatchingDocs.pl?L2/05-151 (accessed 1 May 2017).

[Pournader] Pournader, Roozbeh. 2009. Proposal to encode four combining Arabic characters for Koranic use. http://www.unicode.org/cgi-bin/GetMatchingDocs.pl?L2/09-419 (accessed 2 May 2017).

[Quran1] Quran example. Al-Baqarah. https://archive.org/stream/quran-pdf/002%20-%20Al-Baqarah (accessed 27 Jul 2017).

[Quran2] Quran example. http://www.dailyayat.com/al-ambiya/21/88 (accessed 27 Jul 2017).

[Reports] Unicode Technical Reports
http://www.unicode.org/reports/
For information on the status and development process for technical reports, and for a list of technical reports.

[Unicode] The Unicode Standard
For the latest version, see:
http://www.unicode.org/versions/latest/
For the 10.0.0 version, see:
http://www.unicode.org/versions/Unicode10.0.0/

[Versions] Versions of the Unicode Standard
http://www.unicode.org/standard/versions/
For information on version numbering, and citing and referencing the Unicode Standard, the Unicode Character Database, and Unicode Technical Reports.

Modifications

The following summarizes modifications from the previous revision of this document.

- **Revision 3:**
  - Draft Technical Report

- Summary
  - Summary URL turned into a reference.
- Description of the Algorithm
  - Renamed section.
  - Removed Review Note and incorporated text into following paragraph.
- 3.1 Modifier Combining Marks (MCM)
  - Added U+08D3 ARABIC SMALL LOW WAW to MCM.
- 3.2 Specification of AMTRA
  - Renamed section.
- 4.3 Examples
  - Removed reference to ARABIC NASALIZATION MARK.
  - Changed self reference to [Pournader] to a normal reference.
  - Added rationale for including U+08D3 in MCM.
- 5.6 Other uses for AMTRA
  - Small editorial changes.
- References
  - Added Microsoft Typography 2017 link.

- **Revision 2:**
  - Proposed Draft Technical Report.
  - Converted to html.
  - Renamed algorithm to have a more pronounceable acronym.
  - Title
    - Renamed document.
  - Summary
    - Clarified scope and intent of this UTR.
  - 1. Overview
    - Reworded the Overview for clarity.
  - 3.1 Modifier Combining Marks (MCM)
    - Reworded stability statements in final paragraph.
  - 3.2 AMTRA
    - Reworded steps 2b and 2c for clarity.
  - 4.3 Examples
    - Reworded paragraph in Example 3.
    - Added explanatory details and graphics for Examples 1, 3, 4a and 4b.
  - 5.4 Rationale for exclusion of some marks
    - Reworded first paragraph.
  - 5.6 Other uses for AMTRA
    - Clarified that there is no intention or expectation that AMTRA would be applied to stored text.
    - Expanded to indicate why a text editor may want to utilize AMTRA within backspace processing.
  - References
    - Modified Microsoft Typography link.
    - Sorted per Unicode document conventions.

- **Revision 1:**
  - Initial Draft.
