Technical Reports

L2/26-106

Proposed Update Unicode® Standard Annex #53

UNICODE ARABIC MARK RENDERING

Version	Unicode 18.0.0 (draft 1)
Editors	Roozbeh Pournader (roozbeh@unicode.org), Bob Hallissy (bob_hallissy@sil.org), Lorna Evans (lorna_evans@sil.org)
Date	2026-01-29
This Version	https://www.unicode.org/reports/tr53/tr53-12.html
Previous Version	https://www.unicode.org/reports/tr53/tr53-11.html
Latest Version	https://www.unicode.org/reports/tr53/
Latest Proposed Update	https://www.unicode.org/reports/tr53/proposed.html
Revision	12

Summary

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

This UAX 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*This is a **draft** document which may be updated, replaced, or superseded by other documents at any time.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 Standard Annex (UAX)** forms an integral part of the Unicode Standard, but is published online as a separate document. The Unicode Standard may require conformance to normative content in a Unicode Standard Annex, if so specified in the Conformance chapter of that version of the Unicode Standard. The version number of a UAX document corresponds to the version of the Unicode Standard of which it forms a part.*

Please submit corrigenda and other comments with the online reporting form [Feedback]. Related information that is useful in understanding this document is found in Unicode Standard Annex #41, "Common References for Unicode Standard Annexes." 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]. For any errata which may apply to this annex, see [Errata].

Contents

---

1 Overview

The assignment of Canonical_Combining_Class values for Arabic combining characters in Unicode is different than for most other scripts. It is 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 combining character sequences containing Arabic marks. 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 Section 5.13, *Unknown and Missing Characters, Section 7.9*, *Combining Marks*, and *Section 9.2*, *Arabic* in [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, that is, 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 the Unicode Standard, 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, combining alef overlay
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 except small low noon with kasra
230	All other above combining marks, small low noon with kasra

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)

This specification defines a group of combining marks called “Modifier Combining Marks” (MCM) for use by this algorithm. 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, and so on) in the Unicode Standard, but they are usually perceived by users as *ijam*.

The complete list of MCM characters is defined in the Unicode Character Database (see [UCD]) file PropList.txt.

The set of MCM characters is intended to be 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).

Implementation note: Considering that most Arabic fonts have higher quality glyphs for precomposed characters, implementations may try to recombine base characters with a combining character immediately following them if that would result in a precomposed Unicode character. For example, if after running AMTRA the first two characters of the output are U+064A ARABIC LETTER YEH and U+0654 ARABIC HAMZA ABOVE, an implementation may want to replace them with U+0626 ARABIC LETTER YEH WITH HAMZA ABOVE. (This also helps make sure that the dots of U+064A are not displayed, even if the font is not aware of the Unicode requirement for U+064A losing its dots when combined with U+0654.)

When this step is done, implementations should not skip combining marks. For example, if the output of AMTRA is the sequence <U+0627 ARABIC LETTER ALEF, U+0670 ARABIC LETTER SUPERSCRIPT ALEF, U+0653 ARABIC MADDAH ABOVE>, an implementation may not replace the first and the third character with U+0622 ARABIC LETTER ALEF WITH MADDA ABOVE.

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

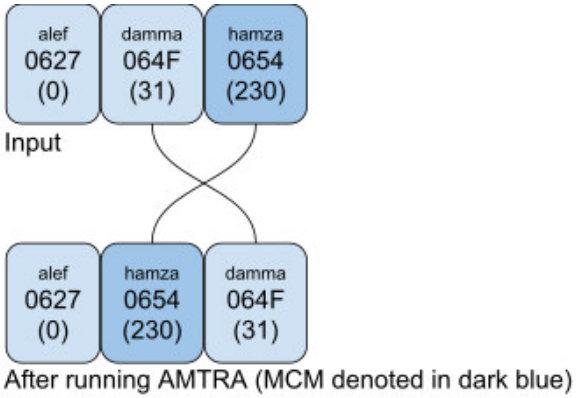
The following examples demonstrate why each of the respective characters is included in the MCM.

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

The use of combining hamza above and below is discussed in *Section 9.2*, *Arabic* in [Unicode].

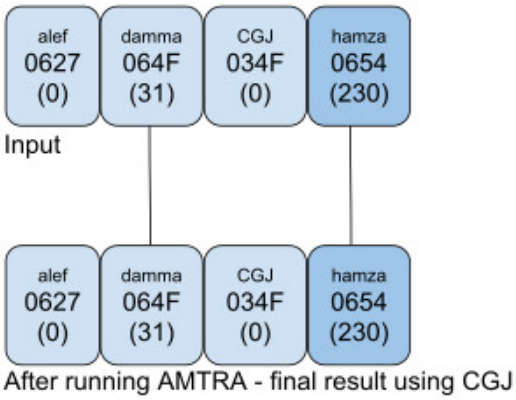
Example 1 [Quran1] (page 9, end of line 5)

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



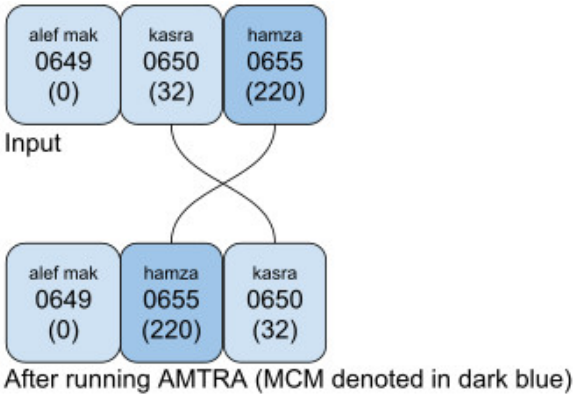
After running AMTRA (MCM denoted in dark blue)

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



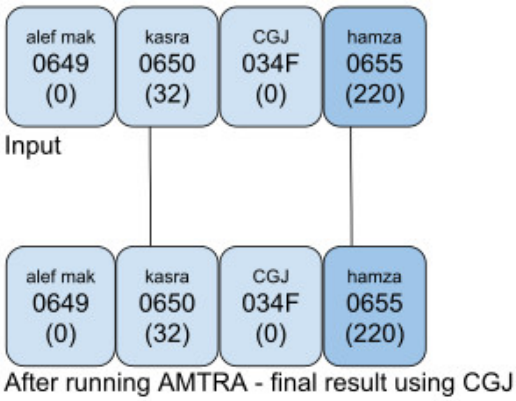
After running AMTRA - final result using CGJ

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



After running AMTRA (MCM denoted in dark blue)

If an orthography needs to place 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

ARABIC 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]

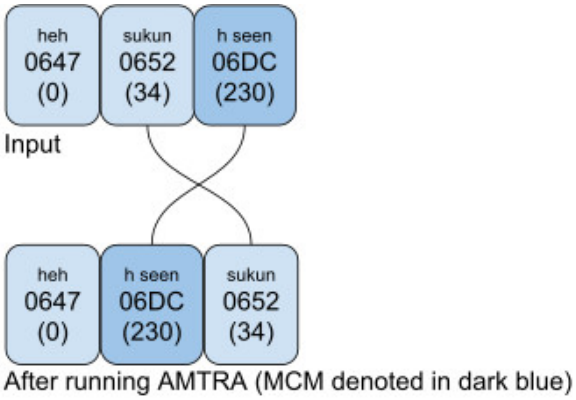
Example 2b [Al-Hilâlî]

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.

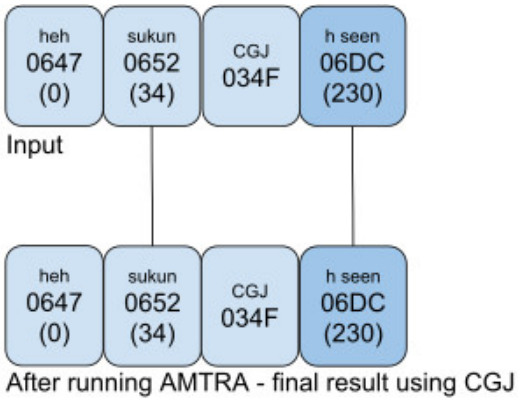
Example 2a (characters and properties):

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

Example 2b (characters and properties):

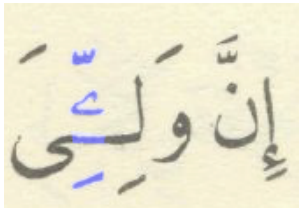


Running AMTRA on the string in Example 2b resulted in an undesired change. It puts a *sukun* over a *seen above*. If an orthography needs to place the *seen above* over the *sukun*, the text should be encoded as <sukun, CGJ, seen above>.

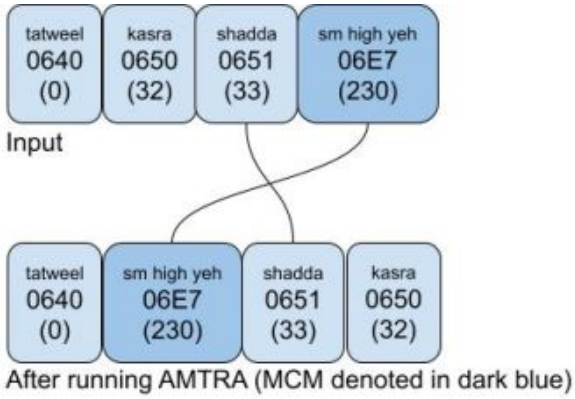


U+06E7 ARABIC SMALL HIGH YEH

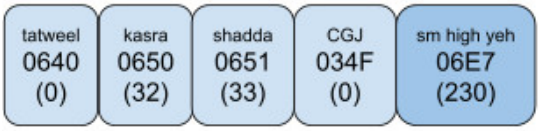
Example 3 [Milo] (page 9, line 11)



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



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



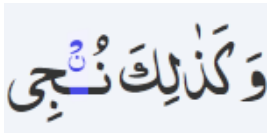
Running AMTRA on this string does not result in any changes.

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.

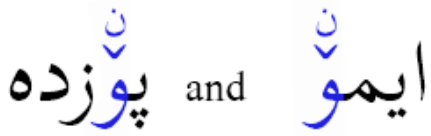
U+06E8 ARABIC SMALL HIGH NOON

Example 4a [Quran2]

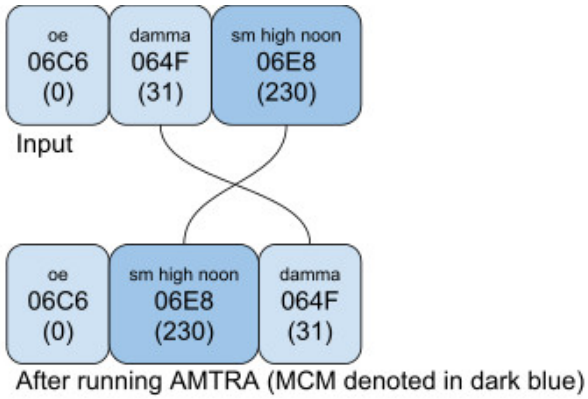


Example 4a has a *sukun* over a *small high noon*. AMTRA puts a *sukun* over a *small high noon*. If an orthography needs to place *small high noon* over *sukun*, the text should be encoded as <sukun, CGJ, small high noon>.

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>).

U+08CE ARABIC LARGE ROUND DOT ABOVE and U+08CF ARABIC LARGE ROUND DOT BELOW

Example 5 [Quran3]

Example 5 has a *fatha* over a *large round dot above*. AMTRA puts a *fatha* over a *large round dot above*. If an orthography needs to place *large round dot above* over *fatha*, the text should be encoded as <fatha, CGJ, large round dot above>.

alef	fatha	CGJ	lg rd dot abv
0627	064E	034F	08CE
(0)	(30)	(0)	(230)

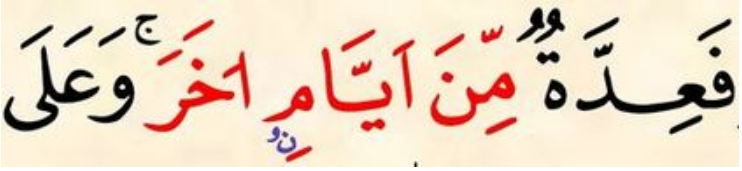
U+10EF4 ARABIC SMALL LOW NOON WITH FATHA and U+10EF6 ARABIC SMALL LOW NOON WITH DAMMA

Example 6a [Quran5]

Example 6a has a *kasra* below a *small low noon with fatha*. AMTRA puts a *kasra* below a *small low noon with fatha*. If an orthography needs to place *small low noon with fatha* below *kasra*, the text should be encoded as <kasra, CGJ, small low noon with fatha>.

teh mrbt	kasra	CGJ	sm low noon fatha
0629	0650	034F	10EF4
(0)	(32)	(0)	(220)

Example 6b [Quran5]



Example 6b has a *kasra* below a *small low noon with damma*. AMTRA puts a *kasra* below a *small low noon with damma*. If an orthography needs to place *small low noon with damma* below *kasra*, the text should be encoded as <kasra, CGJ, small low noon with damma>.

meem	kasra	CGJ	sm low noon damma
0645	0650	034F	10EF6
(0)	(32)	(0)	(220)

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 had used NFC instead, the sequence <alef, superscript alef, madda> would have resulted in a different order of combining marks than <lam, superscript alef, madda>, because NFC composes <alef, madda> to <alef-with-madda-above>.

5.2 Shadda

The Canonical_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

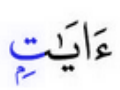
AMTRA is able to handle the special ligation of *kasra* and *kasra-like* characters which are ligated with a *shadda* or *hamza* in some styles and appear just below them instead of below the base letter; they still

logically follow the *shadda* or *hamza*.

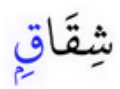
5.4 Rationale for Exclusion of Some Marks

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 6a-6c show that the *combining meem* is normally stored 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*.

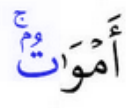
Example 7a [Quran1] (page 11)



Example 7b [Quran1] (page 21)



Example 7c [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, because 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.

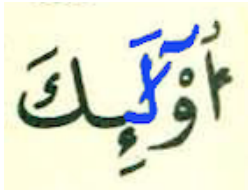
Maddah

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.

Combining Alef Overlay

U+10EFC ARABIC COMBINING ALEF OVERLAY (ccc=0) cannot be MCM because any character with a ccc=0 is not moved by AMTRA. The input for the example below must be *lam*, *fatha*, *alef overlay*, *madda*, and it will not be reordered by the algorithm.

Example 8 [Quran4] (page 502)



5.5 Dotted Circles

Some rendering engines will insert a dotted circle for what they understand 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

AMTRA is not intended or expected to be applied to stored text. However, there may be situations unrelated to rendering where AMTRA may be useful, and this UAX does not prohibit such use.

For 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 expect that the *outermost* marks should be removed first. For Arabic script the AMTRA could be used to identify outermost marks.

5.7 Canonical_Combining_Class values 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 Canonical_Combining_Class as the *tashkil* could be used, but extreme care should be taken.

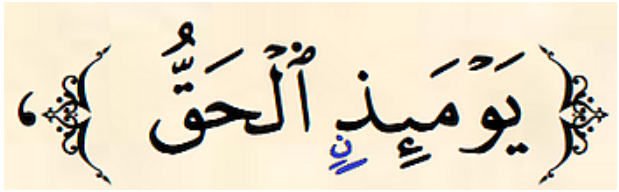
5.8 Workaround for Mistaken Canonical_Combining_Class Assignment

U+08D9 ARABIC SMALL LOW NOON WITH KASRA

When it was added to Unicode 9.0, the *small low noon with kasra* (which appears below the text) was mistakenly given a ccc=230 (mark above). It should have been 220 (mark below), but that cannot now be changed. When used with other combining marks, there are a number of issues:

- When used with any ccc=220 (marks below), in the absence of any *combining grapheme joiner* characters, the reordering by AMTRA will always place the ccc=220 marks between the base character and the *small low noon with kasra*. If this is not desired then a *combining grapheme joiner* can be used to prevent the reordering.
- Combining class sequences containing both U+08D9 and another character of ccc=230 might have the same display but not be canonically equivalent. For this reason it is recommended that U+08D9 be encoded at the very end of the combining mark sequence.
- In Quranic orthographies where U+08D9 appears between a base letter and a *kasra*, such as Example 9 below, a *combining grapheme joiner* must be used in order to control the display order so that U+08D9 is not reordered by AMTRA to move after the *kasra*. The input for Example 9 must be <thal, low noon with kasra, CGJ, kasra>.
- Font developers should make sure that *small low noon with kasra* is treated as if it was a mark below and therefore has no impact on rendering of any marks above.

Example 9 [Quran5] (page 635)



Acknowledgements

Roozbeh Pournader authored the initial concept. Bob Hallissy and Lorna Evans assisted Roozbeh Pournader in turning that concept into this technical report. The three co-authors and co-editors continue to contribute to this document, making various technical and editorial changes, including the classification of newly encoded Arabic combining marks.

Thanks to David Corbett, Behnam Esfahbod, Asmus Freytag, Ned Holbrook, Richard Ishida, Thomas Milo, and Ken Whistler for feedback on and contributions to this document, including earlier versions.

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.

[Kew] Kew, Jonathan, 2002. Bidi committee consensus on Arabic additions from L2/01-425. 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). L2/05-151 (accessed 1 May 2017).

[Pournader] Pournader, Roozbeh. 2010. Proposal to encode four combining Arabic characters for Koranic use. L2/09-419R (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).

[Quran3] Quran example. https://app.quranflash.com/book/Warsh2?en#/reader/chapter/565 (accessed 21 Dec 2020).

[Quran4] Quran example. https://karachvi.com/quran/mushaf-al-jamahiriya.pdf (accessed 9 Nov 2023).

[Quran5] Quran example. http://www.alwa7y.com/downloads/TayseerWarsh.pdf (accessed 21 Jan 2026).

Modifications

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

Revision 12:

- Added to MCM: U+10EF4 ARABIC SMALL LOW NOON WITH FATHA.
- Added to MCM: U+10EF6 ARABIC SMALL LOW NOON WITH DAMMA.
- **Proposed Update** for Unicode 18.0.0

~~**Revision 11:**~~

- ~~**Reissued** for Unicode 17.0.0~~

Modifications for previous versions are listed in those respective versions.

© 2018–2026 Unicode, Inc. This publication is protected by copyright, and permission must be obtained from Unicode, Inc. prior to any reproduction, modification, or other use not permitted by the Terms of Use. Specifically, you may make copies of this publication and may annotate and translate it solely for personal or internal business purposes and not for public distribution, provided that any such permitted copies and modifications fully reproduce all copyright and other legal notices contained in the original. You may not make copies of or modifications to this publication for public distribution, or incorporate it in whole or in part into any product or publication without the express written permission of Unicode.

Use of all Unicode Products, including this publication, is governed by the Unicode Terms of Use. The authors, contributors, and publishers have taken care in the preparation of this publication, but make no express or implied representation or warranty of any kind and assume no responsibility or liability for errors or omissions or for consequential or incidental damages that may arise therefrom. This publication is provided “AS-IS” without charge as a convenience to users.

Unicode and the Unicode Logo are registered trademarks of Unicode, Inc., in the United States and other countries.