Comments on SC2 N4635, CD Text of 10646 6th Edition

Peter Constable, October 20, 2018

This document provides comments on the CD text in SC2 N4635 up through clause 12.

1. (E) — Forward

The following text reflects the fifth edition, not the sixth, and needs to be updated.

> This fifth edition of ISO/IEC 10646 cancels and replaces the fourth edition (ISO/IEC 10646:2014), which has been technically revised. It also incorporates ISO/IEC 10646:2014/Amd 1:2015 and ISO/IEC 10646:2014/Amd 2:2016.
>
> This edition includes the following significant changes with respect to the previous edition:
>
> - New scripts covered: Adlam, Bhaiksuki, , Marchen, Masaram Gondhi, Newa, Nushu, Osage, Soyombo, Tangut, and Zanabazar Square,
> - Existing scripts significantly extended: Cherokee, CJK Unified Ideographs (Extension F),
> - New Emoji symbols.

Change to:

> This sixth edition of ISO/IEC 10646 cancels and replaces the fifth edition (ISO/IEC 10646:2017), which has been technically revised. It also incorporates ISO/IEC 10646:2017/Amd 1:2018 and ISO/IEC 10646:2017/Amd 2:2019.
>
> *other text as the editor deems appropriate*

2. (E) — Clause 1, Scope

The items in the bulleted list are delimited using comma “,”. However, several of the bullet items include expressions separated with commas. Normal editorial convention in such cases is for the higher-level boundaries to be separated using semi-colons. See clause 23 in ISO/IEC Directives Part 2, 8th edition, for an example that illustrates this.

Proposed change: replace commas at the end of bullet items with semi-colon.

3. (E) — Clause 1, Scope

In the bulleted list, the fourth bullet covers the BMP, and the fifth bullet covers the assigned supplementary planes. This fundamental distinction between the BMP and supplementary planes is anachronistic, a hold-over from when there were separate 10646-1 and 10646-2 standards.

Proposed change: Merge the fourth and fifth bullets into one:

- specifies the assigned planes of the UCS: the Basic Multilingual Plane (BMP), the Supplementary Multilingual Plane (SMP), the Supplementary Ideographic Plane (SIP), the Tertiary Ideographic Plane (TIP), and the Supplementary Special-purpose Plane (SSP);

(Proposed text ends with semi-colon, as proposed in comment 2.)

4. (general) — Clause 3, Terms and definitions

Links are provided to terminology databases in IEC Electropedia and ISO OBP. Terms from 10646 do not appear to be included in either of these, however.

5. (E) — Clause 3.10, code point

Add note:

> "Note 1 to entry — Code points in the UCS codespace are integer values. Throughout this document, UCS code points are cited in hexadecimal. UCS code points range from 0 to 10FFFF.

6. (E) — Clause 3.12, code unit sequence

In note 1:

> "… any type of code points."

Change to:

> "… any type of code point."

7. (E) — Clause 3.12, code unit sequence

Note 1 refers to *types* of code points. However, this concept has not yet been introduced.

Proposed change: Add at the end of note 1, "(See clause 6.3.)"

8. (T) — Clause 3.13, collection

The definition given is:

> "numbered and named set of entities"

"Entities" is open-ended — it could include cities, unicorns, chemical formulas, etc. For the UCS context, something specific is intended, but there is no explanation of what that is. *Code points? Code point sequences?*

(Note: see related comment 15, below, for 3.25.)

Proposed change: Add qualifiers in the definition or in a note (new note 1) clarifying what kinds of entities are included in UCS collections. Specific, proposed wording is not provided here since it's not clear what is actually intended.

9. (E) — Clause 3.14, combining character

In note 1 (two occurrences):

> "…non-combining graphic character…"

By 3.1, "non-combining graphic character" is the same as "base character".

Proposed change: replace “non-combining graphic character” with “base character”.

10. (E) — Clause 3.15, combining class

Add note:

> Note 1 to entry — See 20.2 for details on canonical ordering.

11. (E) — Clause 3.17, composite sequence

In the Unicode Standard, the corresponding term is “combining character sequence”.

Proposed change: Add “combining character sequence” as an alternate term for the same concept.

12. (E) — Clause 3.17, composite sequence

The definition includes a cross-reference, “(see also 3.14)”. Cross references should be provided in notes, not in the definition. (Per 16.5.6 of ISO/IEC Directives Part 2, “The definition shall be written in such a form that it can replace the term in its context.”)

Proposed change: Add a new note:

> “Note 1 to entry — See also 3.14.”

Renumber subsequent notes.

13. (E) — Clause 3.20, decomposition mapping

The terminology *canonical equivalent* and *compatibility equivalent* are used but are not defined in clause 3, or anywhere else in the document!

The definition of a term in clause 3 should not depend on terms that are not defined or explained within the doc. However, a thorough explanation would require details within chapter 3 of the Unicode Standard as well as UAX #15.

Proposed change: Replace the definition with the following:

> mapping from a character to a sequence of one or more characters
>
>> Note 1 to entry — Decomposition mappings are of two types: canonical decompositions, and compatibility decompositions. These are used in the derivation of various normalization forms (see 28). The code charts for various blocks include decomposition mappings and distinguish between the two types of mapping (see 33.3).

14. (E) — Clause 3.24, encoding scheme

In note 1:

> “… Some of the UCS encoding schemes have the same labels as the UCS encoding form. However, they are used in different contexts…”

Two issues:

- The definite article in “the UCS encoding form” assumes this is unique and implicitly-understood.

- The antecedent of “they” is unclear: labels for encoding schemes and encoding forms are used in different contexts? Or encoding schemes and encoding forms themselves are used in different contexts?

Proposed change:

> “… Some of the UCS encoding schemes have the same labels as UCS encoding forms. However, references to encoding schemes and encoding forms generally occur in different contexts…

15. (E) — Clause 3.25, extended collection, and note 1 of clause 3.13, collection

The definition of extended collection in 3.25 and the description of non-extended collections in 3.13 are unclear. The definition in 3.25 states (emphasis added),

> “collection for which the entities *can also* consist of sequences of code points that are in Normalization Form C”

This seems to imply that a non-extended collection must not include “sequences of code points that are in Normalization Form C”. That, in turn, seems to imply that a non-extended collection *can* include sequences of code points so long as they are not in Normalization Form C (including sequences that have proper sub-sequences that are in Normalization Form C).

It’s not clear if that is the actual intent, however. In 3.13, non-extended sequences are described as sets that

> “… consist only of those coded characters whose code points lie within one or more identified ranges”

That description is, itself, vague, since *any* set of coded characters code be defined using code points that “lie within one or more identified ranges”, unless some constraint is imposed on “identified ranges”. But it does not seem to correspond in any clear way to the definition in 3.25.

Proposed change: Give clearer definitions. Specific, proposed wording is not provided here since it is not clear what is actually intended.

16. (T) — Clause 3.27, format character

> “character whose primary function is to affect the layout or processing of characters around it”

It is unclear whether the term defined here is intended to have a 1:1 correspondence with the *Format* basic type in the code point classification given in clause 6.3.1. As described in comment 40 below, the class of *Format* characters is somewhat complex: most are invisible controls affecting layout or processing of neighboring characters, but some produce a *visible* presentation, albeit one that involves a complex interaction with neighboring characters.

Proposed change:

> “character whose primary function is to affect the layout or processing of characters around it, or that is presented in a complex, graphic interaction with neighboring characters”

17. (E) — Clause 3.28, General Category

In note 1:

> "Each value is defined as General Category property using a two-letter abbreviation in the Unicode Standard…"

Wording is unclear. Change to:

> "Possible values are two-letter abbreviations defined for the General Category property in the Unicode Standard…"

18. (E) — Clause 3.31, high-surrogate code point

> "code point in the range D800 to DBFF reserved for the use of UTF-16"

Change to:

> "code point in the range D800 to DBFF
>
> > "Note 1 to entry — Reserved for use in UTF-16 (see 9.3)."

19. (E) — Clause 3.32, high-surrogate code unit

> "16-bit code unit in the range D800 to DBFF used in UTF-16 as the leading code unit of a surrogate pair (see 9.3)"

Change to:

> "16-bit code unit in the range D800 to DBFF and used in UTF-16
>
> > "Note 1 to entry — A high-surrogate code unit is used as the leading code unit of a surrogate pair. See also 3.40, 3.55 and 9.3."

20. (E) — Clause 3.34, ill-formed code unit sequence subset

Sets and subsets are not ordered, whereas a sequence is an ordered list of entities. The entities in a sequence can be described as coming from a set, but a sequence can include multiple instances of a given entity, whereas the set does not. Hence, the terminology "sequence subset" is odd and unclear.

This issue arises in the term being defined as well as in the definition.

Also, the definition has restrictive relative clauses introduced using "which", not "that".

Proposed changes:

- Change the term to "ill-formed code unit subsequence" (or "sub-sequence").
- Change the definition to the following.

> "non-empty subsequence of a code unit sequence X that does not contain any code unit that belongs to a minimal well-formed code unit subsequence of X
>
> > "Note 1 to entry — An ill-formed code unit subsequence cannot overlap with a minimal well-formed code unit sequence."

Note: this term is only found in clause 3.61.

21. (E) — Clause 3.36, interworking

This term is not used anywhere else within the document, so it is not clear why the term is defined at all. Per ISO/IEC Directives Part 2, clause 16.5.4, only terms that are used within the document should be included in clause 3.

Proposed change: Delete this clause.

22. (E) — Clause 3.37, ISO/IEC 10646-1

> "… the specification of the overall architecture and the Basic Multilingual Plane (BMP)"

Change to:

> "… the specification of the overall UCS architecture and of the Basic Multilingual Plane (BMP)"

23. (E) — Clause 3.39, low-surrogate code point

> "code point in the range DC00 to DFFF reserved for the use of UTF-16"

Change to:

> "code point in the range DC00 to DFFF
>
> > "Note 1 to entry — Reserved for use in UTF-16 (see 9.3)."

24. (E) — Clause 3.40, low-surrogate code unit

> "16-bit code unit in the range DC00 to DFFF used in UTF-16 as the trailing code unit of a surrogate pair (see 9.3)"

Change to:

> "16-bit code unit in the range DC00 to DFFF and used in UTF-16
>
> > "Note 1 to entry — A low-surrogate code unit is used as the trailing code unit of a surrogate pair. See also 3.32, 3.55 and 9.3."

25. (E) — Clause 3.44, plane

> "subdivision of the UCS codespace consisting of contiguous 65 536 code points beginning at a multiple of 65 536 which can be identified by a number from 00 to 10"

Change to:

> "subdivision of the UCS codespace consisting of 65 536 contiguous code points beginning at a multiple of 65 536
>
> > "Note to entry 1 — UCS planes can be identified by a hexadecimal number from 00 to 10"

26. (E) — Clause 3.49, row

> "subdivision of a plane consisting of contiguous 256 code points beginning at a multiple of 256 which can be identified by a number from 00 to FF"

Change to:

> “subdivision of a plane consisting of 256 contiguous code points beginning at a multiple of 256
>
> > “Note to entry 1 — Within the context of a given place, rows can be identified by a hexadecimal number from 00 to FF.”

27. (E) — Clause 3.52, Supplementary Multilingual Plane for scripts and symbols

The names of planes do not include a description of the characters or blocks within the plane.

Proposed change: Change the term to “Supplementary Multilingual Plane” (i.e., remove “for scripts and symbols”).

28. (E) — Clause 3.55, surrogate pair

> “representation for a single character…”

Change to:

> “UTF-16 encoded representation for a single supplementary-plane character…

29. (E) — Clause 3.59, unpaired surrogate code unit

> “code unit in a code unit sequence…”

Change to:

> “code unit in a UTF-16 code unit sequence…”

Also add a note:

> > “Note 1 to entry — Any unpaired surrogate code unit constitutes an ill-formed code unit sequence.”

30. (E) — Clause 3.61, well-formed code unit sequence

> “… and contains no ill-formed code unit sequence subset”

Change to:

> “… and contains no ill-formed code unit subsequence”

See the related comment 20, above, on clause 3.34.

31. (T) — Clause 4.2, Conformance of information interchange

In list item a),

> “… Clause 0…”

Change to “Clause 6”.

32. (T) — Clause 5, General structure of the UCS

In paragraph 2,

> “… from 0 to 10FFFF.”

Change to:

> “… from 0 to 10FFFF (hexadecimal).”

Note: this change is not needed if the proposed change for clause 3.10 given in comment 5, above, is accepted.

33. (E) — Clause 5, General structure of the UCS

Second item of bulleted list after paragraph 2:

> “The Supplementary Multilingual Plane for scripts and symbols…”

Change to:

> “The Supplementary Multilingual Plane…”

34. (T) — Clause 5, General structure of the UCS

In the paragraph after the bulleted list:

> “The Tertiary Ideographic Plane (TIP, Plane 03) is reserved for ideographic characters and is currently empty.”

As of this CD, the TIP is no longer empty.

Proposed changes:

- Delete this sentence from that paragraph.
- Insert a bullet item in the preceding bulleted list:

> “• The Tertiary Ideographic Plane (TIP, Plane 03).”

35. (E) — Clause 5, General structure of the UCS

In the last paragraph:

> “… coding space…”

Change to:

> “… codespace…”

36. (E) — Clause 6.1, Structure

Figure 1 calls out Supplementary planes. This is anachronistic, a hold-over from when there were separate 10646-1 and 10646-2 standards. (See related comment 3, above.) While the term “supplementary plane” still is useful for 10646, its usefulness is specific to the UTF-16 encoding form and description of the surrogate code points used for UTF-16. In a general description of the structure of the code space, it is better to omit this.

37. (E) — Clause 6.2, Coding of characters

> “Each encoded character within the UCS codespace is represented by an integer between 0 and 10FFFF identified as a code point.”

Proposed change: insert “(hexadecimal)” after “10FFFF”.

Note: this change is not needed if the proposed change for clause 3.10 given in comment 5, above, is accepted.

38. (E) — Clause 6.2, Coding of characters

> “When referring to characters within plane 00, the leading two digits may be omitted; for characters within planes 01 to 0F, the leading digit may be omitted, such as”

The structure of this sentence is such that the examples that follow seems to pertain only to planes 01 to 0F.

Change to:

> “When referring to characters within plane 00, the leading two digits may be omitted; for characters within planes 01 to 0F, the leading digit may be omitted. For example:”

39. (E) — Clause 6.3.1, Classification

> “UCS code points are categorized in basic types…”

Change “in” to “into”

40. (T) — Clause 6.3.1, Classification, Table 1

The stated General Category values for Format are Cf, Zl and Zp, and the brief description says,

> “Invisible, but affects neighbouring characters”

The stated General Category values for Control are Cc, and the brief description says,

> “Control functions consisting of a single code point”

There are some issues with this.

The first issue pertains to Zl and Zp: the code points assigned these values are single code points, and the characters are control functions: LINE SEPARATOR, PARAGRAPH SEPARATOR. Thus, the GC values seem to fit the description currently given for Control.

A comparison with corresponding descriptions in the Unicode Standard may be useful: it gives (Unicode 11.0, Chapter 2, Table 2-3) a different description for Control:

> “Usage defined by protocols or standards outside the Unicode Standard”

A similar description could be equally applied to the description of Control in 10646 since, as noted in clause 11, the semantics for all C0 and C1 controls and for DELETE are specified in a different standard, ISO/IEC 6429. The LINE SEPARATOR and PARAGRAPH SEPARATOR control functions, however, are defined in 10646, not in ISO/IEC 6429. Thus, if the description for Control were similar to that used in Unicode, distinguishing functions defined outside rather than within the current document, then Zl and Zp would *not* fit that description for Control.

Proposed change: Change the brief description for Control to:

> “Control functions defined in ISO/IEC 6429”

or to:

> “Control functions defined outside this document”

The second issue pertains to the description for Format and characters that have General Category of Cf: Most Cf characters are invisible; for example:

- 061C ARABIC LETTER MARK
- 180E MONGOLIAN VOWEL SEPARATOR
- 200B ZERO WIDTH SPACE
- 200C ZERO WIDTH NON-JOINER
- 200D ZERO WIDTH JOINER
- 200E LEFT-TO-RIGHT MARK
- 200F RIGHT-TO-LEFT MARK
- etc.

However, at least one Cf character might be considered *conditionally invisible* — that is, invisible in some contexts, but given a visual presentation in other contexts:

- 00AD SOFT HYPHEN

Moreover, several other Cf characters always produce a visual presentation, albeit one that involves some complex interaction with neighboring characters; for example:

- 0600 ARABIC NUMBER SIGN
- 0601 ARABIC SIGN SANAH
- 0602 ARABIC FOOTNOTE MARKER
- 0603 ARABIC SIGN SAFHA
- 0604 ARABIC SIGN SAMVAT
- 0605 ARABIC NUMBER MARK ABOVE
- 06DD ARABIC END OF AYAH
- 070F SYRIAC ABBREVIATION MARK
- 08E2 ARABIC DISPUTED END OF AYAH
- 110BD KAITHI NUMBER SIGN
- 110CD KAITHI NUMBER SIGN ABOVE

At a minimum, then, “invisible” in the description for Format seems to be inappropriate.

It is also worth noting that the definition of *format character* in clause 3.27 mentioned “layout or processing”.

Proposed change: Change the brief description for Format to:

> “Affects layout or processing of neighboring characters, or has a complex graphic interaction with neighboring characters”

41. (T) — Clause 6.3.3, Format characters

As described in comment 40, the class of Format characters is complex in that some are invisible controls while others are not. The following is proposed as a replacement for the current text:

> Format characters form a class of characters that affect or strongly interact with neighbouring characters.

Most format characters are invisible but affect the layout or processing of neighboring characters. For example:

061C	ARABIC LETTER MARK
200B	ZERO WIDTH SPACE
2062	INVISIBLE TIMES

Some format controls have a visible presentation, but one that involves a complex graphic interaction with neighboring characters. For example:

0600	ARABIC NUMBER SIGN
070F	SYRIAC ABBREVIATION MARK

42. (E) — Clause 6.3.5, Private use characters

"All code points of Plane 0F and Plane 10, except for FFFFE, FFFFF, 10FFFE, and 10FFFF are reserved for private use."

Wording could be clearer. Change to:

> "All code points of Plane 0F and Plane 10 — with the exception of noncharacter code points FFFFE, FFFFF, 10FFFE, and 10FFFF — are reserved for private use."

43. (E) — Clause 6.3.7, Noncharacter code points

'Code point FFFE is reserved for "signature".'

The term "signature" is nowhere defined, and the meaning and purpose of this statement unclear. There should also be a cross-reference to clause 10, in which context signatures are relevant.

Proposed change: Create a separate note for discussion of FFFE, with wording as follows:

> Code point FFFE is reserved for use as a "signature" for detecting correct byte order in UTF-16 or UTF-32 text data streams. Because the UTF-16 and UTF-32 encoding forms use 16-bit and 32-bit code units, but many processes handle data streams as byte sequences, the ordering of bytes within UTF-16 or UTF-32 code units strongly affects the interpretation of text data streams encoded in these encoding forms. The character FEFF ZERO WIDTH NO-BREAK SPACE, which has a minimal effect on the meaning or processing of text, can be included in a text data stream. If the bytes for this character were interpreted using the wrong byte order, then the bytes would be interpreted as the noncharacter code point FFFE. Since inclusion of this noncharacter code point is not expected in valid text content, the process would be able recognize the correct byte order. See also 10.

44. (E) — Clause 6.4, Naming of characters

"Some characters may have one or more alternate names called character name aliases which are correction of the original names. Additional rules to be used for constructing the names of characters are given in 26."

Proposed change:

> "Some characters may have one or more alternate names, called character name aliases, which are corrections of the original names.

> > “NOTE — Character name aliases, which are normative, should not be confused with informative aliases, which are other names for characters that may be used outside this document but that are not normative.
>
> “Additional rules to be used for constructing the names of characters are given in 26.”

45. (T) — Clause 6.5, Short identifiers for code points (UIDs)

The alternative form as stated in b) is:

> “The four-to-five-digit form of short identifier shall consist of the last four to five digits of the six-digit form. Leading zeroes beyond four digits are suppressed.”

This is problematic in that, as stated, it allows for (e.g.) “2456” to be treated as a short form identifier for the code point 012345. While it states that leading zeroes are suppressed, it does not require that all non-zero digits must be maintained.

Proposed change:

> “The four-to-five-digit form of short identifier shall consist of the last four to five digits of the six-digit form, with all non-zero digits kept but any leading zeroes beyond four digits suppressed.”

46. (E) — Clause 6.6, UCS Sequence Identifiers

> “The UCS Sequence Identifier includes at least two UIDs…”

Change to:

> “A UCS Sequence Identifier must include at least two UIDs…”

47. (T) — Clause 6.6, UCS Sequence Identifiers

The first paragraph describes the elements within a USI as UIDs with “[t]he syntax for UID1, UID2, etc…. specified in 6.5.” The syntax in 6.5 allows for different forms — 017F, +017F, etc. However, the BNF given later in 6.6 only allows for one form for citation of UIDs: hex digits without a prefixed “U” or “+”.

If the intent is that a valid USI can use any of the valid forms for UIDs, then the BNF in clause given later in 6.6 is incorrect. But if the intent is that a valid USI can only use the forms for UIDs that exclude “U” and “+”, then this should be stated in the first paragraph.

If the different forms for UIDs are permitted in USIs, then (depending on the BNF) that could allow for USIs that mix different forms for UIDs; e.g., “<0041, U+0301>”. If use of different UID forms is valid, then a note can be added that a USI should consistently use the same UID form for all UIDs.

Proposed changes: This depends upon what has been the intent.

- If alternate UID forms are to be permitted:
    - Change the BNF in 6.5 to include “UID” as a label for the UID (“UID ::= …”), then use “UID” in the BNF in 6.6 to replace both occurrences of “(xxxx|xxxxx|xxxxxx)”.
    - Add a note recommending that citations for USIs consistently use the same citation form for UIDs.

- If alternate UID forms are not to be permitted: Append clarification on UID forms in the fifth sentence of the first paragraph, as follows:

  > 'The syntax for UID1, UID2, etc. is specified in 6.5, with an added constraint that, within a USI, the UIDs must not include a "U", "u" or "+" prefix.'

48. (T) Clause 8.3, Selected subset

> "A selected subset shall always automatically include the code points from 0020 to 007E."

It's unclear whether "shall always automatically include" is intended to say that 0020 – 007E are always implicitly included, or that this range must be explicitly included for a subset specification to be valid.

For example, suppose that a device is described as supporting a subset specified as "collection 2 (LATIN-1 SUPPLEMENT)". Which of the following is the intent of 8.3?

- The subset specification is interpreted as comprising the code points {0020 – 007E, 00A0 – 00FF}.
- The subset specification is invalid (since collection 1 is not explicitly included in the specification).

The wording seems more likely to mean the former. If so, the following is proposed wording to replace the sentence quoted above:

> "A selected subset shall always automatically and implicitly include the code points from 0020 to 007E, even if the corresponding collection, collection 1 BASIC LATIN, is not explicitly listed in the subset specification."

49. (E) Clause 9.2 UTF-8

> "It indicates the number of continuing octets…"

The antecedent of "it" is unclear: "first octet"? "code unit sequence"? "arbitrary location"?

Change to:

> "The first code unit indicates the number of continuing octets…"

50. (E) Clause 9.2 UTF-8

> "Table 2 specifies the bit distribution for the UTF-8 encoding form, showing the ranges of UCS scalar values…"

Table 2 doesn't clearly show scalar-value ranges; it shows bit distributions, from which a reader could derive scalar-value ranges with some effort.

Proposed change: add a column on the left of Table 2 to show scalar-value ranges:

Scalar values	Scalar value bit distribution	1st octet	2nd octet	3rd octet	4th octet
0000 to 007F	000000000xxxxxxx	0xxxxxxx			
0080 to 07FF	00000yyyyyxxxxxx	110yyyyy	10xxxxxx		
0800 to D7FF E000 to FFFF	zzzzyyyyyyxxxxxx	1110zzzz	10yyyyyy	10xxxxxx	
10000 to 10FFFF	000uuuuuzzzzyyyyyyxxxxxx	11110uuu	10uuzzzz	10yyyyyy	10xxxxxx

51. (T) Clause 9.2 UTF-8

> "Because of the well-formedness conditions specified in table 9.2, the following octet values are disallowed in UTF-8: C0-C1, F5-FE."

The octet value FF should also be disallowed, but it is not mentioned.

Change "F5-FE" to "F5-FF".

52. (E) Clause 9.3 UTF-16

> "UTF-16 optimizes the representation of characters in the BMP which contains the vast majority of common use characters."

The use of symbols in the SMP as emoji has to some degree shifted the balance of "common use" between BMP and SMP such that "vast majority" may be somewhat overstated.

Proposed change: delete the word "vast".

53. (E) Clause 9.4 UTF-16

Proposed change: add an extra column to Table 4 to show scalar-value ranges (as proposed for Table 2 in comment 50). Also, separate 1st and 2nd code units into separate columns (parallel to Table 2).

Scalar values	Scalar value bit distribution	1st code unit	2nd code unit
0000 to D7FF, E000 to FFFF	xxxxxxxxxxxxxxxx	xxxxxxxxxxxxxxxx	
10000 to 10FFFF	000uuuuuxxxxxxxxxxxxxxxx	110110wwwwxxxxxx	110111xxxxxxxxxx

54. (E) Clause 9.5 UTF-32 (UCS-4)

Just as "UCS-2" is obsolete and anachronistic, the term "UCS-4" is now superfluous and only serves to create potential ambiguity.

Proposed changes: remove references to UCS-4 in the primary text but add a note for historical continuity. The combined changes would be to revise clause 9.5 as follows:

> Clause 9.5 UTF-32
>
> UTF-32 is the UCS encoding form that assigns each UCS scalar value to a single unsigned 32-bit code unit.
>
>> NOTE — Former editions of this document included "UCS-4" as an alternate term synonymous with "UTF-32". Use of the term "UCS-4" to refer to this encoding form is deprecated.
>
> Because surrogate code points are not UCS scalar values, UTF-32 code units in the range 0000 D800-0000 DFFF are ill-formed.

55. (E) Clause 10 UCS Encoding schemes

The capitalization in the heading is inconsistent with conventions. (Compare with clause 9, "UCS encoding forms".)

Change "Encoding" to "encoding".

56. (E) Clause 10.1 General

The discussion of signatures should cross-reference clause 6.3.7.

Proposed change: Add at the end of paragraph 1, “(See also 6.3.7.)”

57. (E) Clause 10.3 UTF-16BE

> “The UTF-16BE encoding scheme serializes a UTF-16 code unit sequence by ordering octets in a way that the more significant octet precedes the less significant octet (also known as big-endian ordering).”

A minor wording change is proposed:

> “The UTF-16BE encoding scheme serializes a UTF-16 code unit sequence by ordering the octets for each code unit such that the more significant octet precedes the less significant octet (also known as big-endian ordering).”

58. (T) Clause 10.3 UTF-16BE

Note: This comment and comments 60, 62 and 64 each pertain to discussion of signatures. Each proposes similar text added in clauses 10.3, 10.4, 10.6 and 10.7. An alternate approach would be to add an additional sub-clause 10.9 “Use of signatures” that contains information along the lines proposed by these comments.

This clause indicates how an initial octet sequence <FE FF> would be interpreted as ZERO WIDTH NO-BREAK SPACE and not as a signature, but there is no positive statement regarding octet sequences that are treated as a signature.

Note: see also comment 43 above.

Proposed change: add a note after paragraph 2:

> NOTE — Because the code point FFFE is a noncharacter code point (see 6.3.7), the octet sequence <FF FE> is not valid in the UTF-16BE encoding scheme. If a data stream is assumed to be using the UTF-16BE encoding scheme, an initial octet sequence <FF FE> would serve as a signature strongly suggesting that the assumption of UTF-16BE is invalid. If a data stream is assumed to be using the UTF-16 encoding form but the encoding scheme has not been specified, initial octet sequences <FE FF> or <FF FE> can be used as a heuristic: <FE FF> would strongly suggest that UTF-16BE can be assumed, while <FF FE> would strongly suggest that UTF-16BE should not be assumed. (See also 10.5.)

59. (E) Clause 10.4 UTF-16LE

> “The UTF-16LE encoding scheme serializes a UTF-16 code unit sequence by ordering octets in a way that the less significant octet precedes the more significant octet (also known as little-endian ordering).”

A minor wording change is proposed:

> “The UTF-16LE encoding scheme serializes a UTF-16 code unit sequence by ordering the octets for each code unit such that the less significant octet precedes the more significant octet (also known as little-endian ordering).”

60. (T) Clause 10.4 UTF-16LE

As for comment 58, there is no positive statement regarding octet sequences that are treated as a signature.

Proposed change: add a note after paragraph 2:

> NOTE — Because the code point FFFE is a noncharacter code point (see 6.3.7), the octet sequence <FE FF> is not valid in the UTF-16LE encoding scheme. If a data stream is assumed to be using the UTF-16LE encoding scheme, an initial octet sequence <FE FF> would serve as a signature strongly suggesting that the assumption of UTF-16LE is invalid. If a data stream is assumed to be using the UTF-16 encoding form but the encoding scheme has not been specified, initial octet sequences <FF FE> or <FE FF> can be used as a heuristic: <FF FE> would strongly suggest that UTF-16LE can be assumed, while <FE FF> would strongly suggest that UTF-16LE should not be assumed. (See also 10.5.)

61. (E) Clause 10.6 UTF-32BE

> “The UTF-32BE encoding scheme serializes a UTF-32 code unit sequence by ordering octets in a way that the more significant octets precede the less significant octets (also known as big-endian ordering).”

A minor wording change is proposed:

> “The UTF-32BE encoding scheme serializes a UTF-32 code unit sequence by ordering the octets for each code unit such that the more significant octets precede the less significant octets (also known as big-endian ordering).”

62. (T) Clause 10.6 UTF-32BE

As for comment 58, there is no positive statement regarding octet sequences that are treated as a signature.

Proposed change: add a note after paragraph 2:

> NOTE — Because the code point FFFE is a noncharacter code point (see 6.3.7), the octet sequence <00 00 FF FE> is not valid in the UTF-32BE encoding scheme. If a data stream is assumed to be using the UTF-32BE encoding scheme, an initial octet sequence <00 00 FF FE> would serve as a signature strongly suggesting that the assumption of UTF-32BE is invalid. If a data stream is assumed to be using the UTF-32 encoding form but the encoding scheme has not been specified, initial octet sequences <00 00 FE FF> or <00 FF FE> can be used as a heuristic: <00 00 FE FF> would strongly suggest that UTF-32BE can be assumed, while <00 00 FF FE> would strongly suggest that UTF-32BE should not be assumed. (See also 10.8.)

63. (E) Clause 10.7 UTF-32LE

> “The UTF-32LE encoding scheme serializes a UTF-32 code unit sequence by ordering octets in a way that the less significant octets precede the more significant octets (also known as little-endian ordering).”

A minor wording change is proposed:

> "The UTF-32LE encoding scheme serializes a UTF-32 code unit sequence by ordering the octets for each code unit such that the less significant octets precede the more significant octets (also known as little-endian ordering)."

64. (T) Clause 10.7 UTF-32LE

As for comment 58, there is no positive statement regarding octet sequences that are treated as a signature.

Proposed change: add a note after paragraph 2:

> NOTE — Because the code point FFFE is a noncharacter code point (see 6.3.7), the octet sequence <00 00 FE FF> is not valid in the UTF-32LE encoding scheme. If a data stream is assumed to be using the UTF-32LE encoding scheme, an initial octet sequence <00 00 FE FF> would serve as a signature strongly suggesting that the assumption of UTF-32LE is invalid. If a data stream is assumed to be using the UTF-32 encoding form but the encoding scheme has not been specified, initial octet sequences <00 00 FF FE> or <00 00 FE FF> can be used as a heuristic: <00 00 FF FE> would strongly suggest that UTF-32LE can be assumed, while <00 00 FE FF> would strongly suggest that UTF-32LE should not be assumed. (See also 10.8.)

65. Clause 11 Use of control functions with the UCS

The notation used in escape sequences, such as "02/00" or "02/15" (as seen in clause 12.1), may be unfamiliar and confusing to readers. (Does this derive from ISO/IEC 2022?)

Proposed change: Add a note after the fourth or fifth paragraph explaining the notation and its origin; re-number subsequent notes in this sub-clause.

66. (E) Clause 12 Declaration of identification of features

The wording "declaration of identification of features" is very unclear. Moreover, throughout this clause, "identification" is used inconsistently in ways that are sometimes very unclear and confusing. For example, "the identification of this document" (in 12.1) is not actually referring to an identifier for a document entity but rather is referring to *encoding characteristics* attributed to a document entity (viz., that the given document uses UCS for the encoded representation of text).

More appropriate uses of "identifier" might be "encoding form identifier" or "encoding scheme identifier". But for some instances of "identifier" or "identification" in clause 12, "declaration" would be a more-appropriate term.

It is also noted that some parts of the current text use "designation".

This comment proposes a change to the heading for clause 12. Other related comments pertaining to portions of sub-clauses will be provided below.

Proposed change: replace the heading for clause 12 with the following:

> "Specification of text-encoding attributes"

or:

> "Specification of text-encoding characteristics"

67. (E) Clause 12.1 Purpose and context of identification

In paragraph 1:

> “Code unit sequences conforming to this document are intended to form all or part of a composite unit of coded information that is interchanged between an originator and a recipient. The identification of this document…”

The antecedent of “this document” in the second sentence is ambiguous. In general, this wording is used as a self-reference to this standard. That usage occurs in the first and last sentences of paragraph 1, and by comparing with the 4th edition, it appears that multiple instances of “this International Standard” have been replaced with “this document”, including in the second sentence. However, given the current wording of this paragraph, the occurrence in second sentence appears to be referring to the “composite unit of coded information”.

Proposed change: This issue is intertwined with the issues raised in comment 66. A proposed change for this issue combined with the other issues is provided in the following comment.

68. (E) Clause 12.1 Purpose and context of identification

This comment pertains to and is a continuation of general issues raised in comment 66.

Proposed changes:

- Replace the heading for clause 12.1 with the following:

  “Purpose and context for specification of text-encoding characteristics”

- Replace the content of clause 12.1 with the following — this also incorporates a change related to the issue raised in comment 67:

  > A declaration of the UCS as the encoded representation for text within coded information should also be available to the recipient, along with declarations of the encoding form and encoding scheme adopted by the originator, and possibly also declarations regarding any subsets of the UCS character repertoire that are used. The route by which such declarations are communicated to the recipient is outside the scope of this document.
  >
  > However, some standards for interchange of coded information may permit, or require, that the coded representation of text be declared as part of the interchanged information, and moreover that specification of the text-encoding characteristics be declared using particular coded representations for those declared characteristics. Clause 12 specifies coded representations for declaration of various text-encoding characteristics applicable to the UCS. This includes declarations to specify encoding schemes, to specify graphic character subsets, or to specify control character subsets. Such coded representations provide all or part of a declarative data element, which may be included in information interchange in accordance with the relevant information-interchange standard.

> For the contexts in which such declarations are used, it is assumed that more significant octets in a coded representation shall precede the less significant octets when serialized. For this reason, the only encoding schemes that can be specified using the coded representations provided here are UTF-8, UTF-16BE, and UTF-32BE according to the relevant encoding forms (UTF-8, UTF-16, and UTF-32 respectively).
>
> If two or more of the text-encoding specification are present within a declarative data element, the order of those specifications shall follow the order as specified in Clause 12.
>
>> NOTE – An alternative method for specification of text-encoding characteristics is described in Annex N.

69. (E) Clause 12.1 Purpose and context of identification

In paragraph 4:

> "the order of those [specifications] shall follow the order as specified in Clause 12."

It's not clear what is meant by "the order as specified in clause 12". Does this mean that the required ordering for different kinds of declaration must corresponds to the order of sub-clauses in which the different kinds are described (e.g., that a specification for encoding scheme must precede a specification of a graphic character subset because clause 12.2 precedes clause 12.3)?

Proposed change: Revise the wording to clarify the intended meaning. (Specific wording is not proposed since it's not clear what is intended.)

70. (E) Clause 12.2 Identification of a UCS encoding scheme

This comment pertains to and is a continuation of general issues raised in comment 66.

Proposed changes:

- Replace the heading for clause 12.2 with the following:

  "Specification of a UCS encoding scheme"

- Revise paragraph 1 as follows:

  "When the escape sequences from ISO/IEC 2022 are used, specification of a UCS encoding scheme (see Clause 10) defined by this document shall be by a specification sequence chosen from the following list:"

- In the note, replace "designation sequences" with "specification sequences".

Alternately, "encoding-scheme identifier sequence" could be used instead of "specification sequence".

71. (E) Clause 12.3 Identification of subsets of graphic characters

This comment pertains to and is a continuation of general issues raised in comment 66.

Proposed changes:

- Replace the heading for clause 12.3 with the following:

  "Specification of graphic character subsets"

- Revise paragraph 1 as follows (this does not reflect the issue raised in the following comment):

  "When the control sequences of ISO/IEC 6429 are used, specification of graphic character subsets (see Clause 8) defined by this document shall be by a control sequence IDENTIFY UNIVERSAL CHARACTER SUBSET (IUCS) as shown below."

72. (T) Clause 12.3 Identification of subsets of graphic characters

In paragraph 1:

> "When the control sequences of ISO/IEC 6429 are used…"

Is "control sequences of ISO/IEC 6429" actually what is intended here? Or should this be "escape sequences of ISO/IEC 2022"?

Proposed change: Reference the appropriate standard.

73. (T) Clause 12.3 Identification of subsets of graphic characters

In paragraph 1:

> "… by a control sequence IDENTIFY UNIVERSAL CHARACTER SUBSET (IUCS) as shown below."

This is followed by:

> "CSI Ps... 02/00 06/13"

It's not clear if "CSI" is referring to 6429 control function "009B CONTROL SEQUENCE INTRODUCER" (cited in clause 11). In any case, there is not clear continuity between the reference to IUCS in paragraph 1 and the elaboration that follows.

Proposed change: Provide a clearer specification. (Specific changes are not proposed since it is unclear what precisely is intended.)

74. (E) Clause 12.3 Identification of subsets of graphic characters

In paragraph 3:

> "The parameters are to be taken from the subset collection numbers as shown in 9."

It's unclear what "9" refers to. Given the context involving subsets, it appears that clause 8 may perhaps be intended. It's not clear that Clause 8 would be appropriate, however, since it allows for means of describing subsets that may not be amenable to representation in escape sequences. So, perhaps Annex A is what is intended.

(It appears that the reference to "9" was introduced in the CD for the 4th edition, and that prior to that the text referenced Annex A.)

Proposed change: provide the correct reference. If clause 8 is intended, use “Clause 8” (rather than simply “8”) for clarity.

75. (E) Clause 12.4 Identification of control function set

This comment pertains to and is a continuation of general issues raised in comment 66.

Proposed changes:

- Replace the heading for clause 12.4 with the following:

  “Specification of control function set”

- Revise paragraph 1 as follows:

  “When the escape sequences from ISO/IEC 2022 are used, the specification of each set of control functions (see Clause 11) of ISO/IEC 6429 to be used in conjunction with ISO/IEC 10646 shall be a specification sequence of the type shown below.”

- Revise the second sentence of paragraph 3 as follows:

  “The specification sequences for these sets shall be”

Alternately, “control-function-set identifier” could be used instead of “specification sequence”.

76. (E) Clause 12.5 Identification of the coding system of ISO/IEC 2022

This comment pertains to and is a continuation of general issues raised in comment 66.

Proposed changes:

- Replace the heading for clause 12.4 with the following:

  “Specification of the ISO/IEC 2022 coding system”

- Revise the first sentence of paragraph 1 as follows:

  “When the escape sequences from ISO/IEC 2022 are used, specification of a return, or transfer, from UCS to the coding system of ISO/IEC 2022…”

77. (general) Clause 12 Declaration of identification of features

As suggested by comments 66 through 76, the current state of Clause 12 has several issues. It serves no purpose to provide a specification that is unclear and ambiguous. Many of the issues raised in these comments are very old and have remained through several editions without correction. If it has not been important through all this time to provide a clearer specification, then perhaps that is an indication that clause 12 is no longer important and can be removed.