|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  37th Meeting: Geneva, CH, 4–10 October 2019 | Document: JCTVC-AK1004-v1 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Errata report items for HEVC, AVC, and Video CICP** | | |
| *Status:* | Output document approved by JCT-VC | | |
| *Purpose:* | Errata report | | |
| *Author(s) or Contact(s):* | **Gary Sullivan**  **Ye-Kui Wang** | Tel: Email: | [garysull@miscrosoft.com](mailto:garysull@miscrosoft.com)  [yekui.wang@bytedance.com](mailto:yekui.wang@bytedance.com) |
| *Source:* | Editors | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

This document contains a list of reported errata items for HEVC, AVC, and Video CICP, for tracking purposes. Some of the items have been confirmed by the JCT-VC and have been agreed to require fixing, while some other items have not yet been confirmed.

# General aspects and issues affecting multiple standards

## General and minor matters

Please note that the alignment between the text published by ITU-T and that published by ISO/IEC should also be checked. Minor editorial issues and aspects that are highlighted for potential further checking include the following:

* (for HEVC only) In the semantics of matrix\_coeffs, there seems to be a problem in the HEVC spec (the AVC text is OK), saying "one or more" instead of "both" regarding the following constraint:

matrix\_coeffs shall not be equal to 0 unless both of the following conditions are true:

– BitDepthC is equal to BitDepthY.

– chroma\_format\_idc is equal to 3 (the 4:4:4 chroma format).

* (for HEVC only, in the ISO/IEC version only– this has been corrected in the ITU-T 2019-11 pre-published text) For HEVC, add a missing bibliography entry for ITU-R BT.2035.
* (for HEVC only, in the ISO/IEC version only – this has been corrected in the ITU-T 2019-11 pre-published text) For HEVC, in the semantics of current\_frame\_is\_frame0\_flag (subclause D.3.16), there are two instances of "frame\_packing\_arrangement" that should be "frame\_packing\_arrangement\_type" (in the paragraph describing current\_frame\_is\_frame0\_flag).
* (for HEVC only, in the ISO/IEC version only, if applicable – this has been corrected in the ITU-T 2019-11 pre-published text) For HEVC, add a missing line break between the Bibliography references to ISO/IEC 11664-3 and ISO/IEC 14496-12.
* Updating the reference to Rec. ITU-R BT.2100 (the current version being BT.2100-2) and the associated equations for the ICTCP matrix coefficients interpretation for HLG (esp. check Video CICP; this is correct in Rec. ITU-T H.265 2019-06).
* (for Video CICP only, in the ISO/IEC version only) Logical indentation nesting in the ISO/IEC version of the Video CICP colour interpretation was inadvertently removed in the publication process.
* (for Video CICP only) The range of values for the extended aspect ratio indicator in Video CICP is not clearly specified and may implicitly be interpreted as inadequate to cover the range of values expressed in the video coding standards.

## sYCC colour indicator interpretation

### Status

This item has been confirmed by the JCT-VC and resolved for the ITU-T text, but has not yet been resolved for the ISO/IEC text. It affects multiple standards: HEVC, AVC, and Video CICP (and JPEG XR). For background, see [JCTVC-AJ0023](http://phenix.it-sudparis.eu/jct/doc_end_user/current_document.php?id=10941).

### Description of the issue

This issue relates to the transfer characteristics and matrix coefficients indicators for the sYCC colour representation specified in IEC 61966-2-1. The the semantics of transfer characteristics (Table E.4 of HEVC), and matrix coefficients (Table E.5 of HEVC) need correction to address the issue.

The issue is a bit complicated because the same transfer characteristics indicator value is used for both sRGB and sYCC, but IEC 61966-2-1 actually indicates that the transfer characteristics function should be somewhat different for the two cases. In the sRGB case, the range of the input value is constrained to be from 0 to 1, but in the sYCC case, this constraint should not apply.

The agreed correction for this aspect is to condition the interpretation of the transfer\_characteristics syntax element for the value 13 on the value of the matrix\_coeffs syntax element (since that value would differ between sYCC and sRGB).

Additionally, the the informative remark relating to the matrix coefficients indicator value for IEC 61966-2-1 sYCC should be changed to indicate that sYCC should be indicated with the matrix coefficients indicator equal to 5 (as for Rec. ITU R BT.601) rather than 1 (as for Rec. ITU-R BT.709).

*In E.3.1 (VUI semantics), in Table E.4 (Transfer characteristics interpretation using the transfer\_characteristics syntax element) replace the row for the value 13 with:*

|  |  |  |
| --- | --- | --- |
| 13 | – If matrix\_coeffs is equal to 0           V = *α* \* Lc( 1 ÷ 2.4 ) − ( *α* − 1 ) for 1 >= Lc >= *β*           V = 12.92 \* Lc for *β* > Lc >= 0  – Otherwise           V = *α* \* Lc( 1 ÷ 2.4 ) − ( *α* − 1 ) for Lc >= *β*           V = 12.92 \* Lc for *β* > Lc > −*β*           V = − *α* \* ( −Lc )( 1 ÷ 2.4 ) + ( *α* − 1 ) for −*β* >= Lc | IEC 61966-2-1 sRGB (with matrix\_coeffs equal to 0)  IEC 61966-2-1 sYCC (with matrix\_coeffs equal to 5) |

*In E.3.1 (VUI semantics), after Equation E-3, replace the next paragraph and associated bullet points with:*

In this case, the range of E′R, E′G, and E′B is specified as follows:

– If transfer\_characteristics is equal to 11 or 12, or transfer\_characteristics is equal to 13 and matrix\_coeffs is not equal to 0, E′R, E′G, and E′B are real numbers with values that have a larger range than the range of 0 to 1, inclusive, and their range is not specified in this Specification.

– Otherwise, E′R, E′G and E′B are real numbers in the range of 0 to 1.

*In E.3.1 (VUI semantics), in Table E.5 (Matrix coefficients interpretation using the matrix\_coeffs syntax element), move “*IEC 61966-2-1 sYCC*” from the row for the value 1 to the row for the value 5.*

# Reported errata items for HEVC

See also section 1.

## Publication status background

Rec. ITU-T H.265

* (02/18, Edition 5) Approved 2018-02-13, published 2018-05-11
* (06/19, Edition 6) Approved 2019-06-29, published 2019-09-23
* (10/19, Edition 7) Approved 2019-11-29, pre-published 2019-12-06

ISO/IEC 23008-2

* ISO/IEC 23008-2:2017 (Edition 3); published 2017-10
* ISO/IEC 23008-2:2017/Amd 1:2018 (Additional colour representation code point); published 2018-03
* ISO/IEC 23008-2:2017/Amd 2:2018 (Main 10 still picture profile); published 2018-03
* ISO/IEC 23008-2:2017/Amd 3:2018 (Additional supplemental enhancement information); published 2018-07
* ISO/IEC DIS 23008-2:201x (Edition 4); DIS ballot closed 2018-10-06; stage 40.99, Full report circulated: DIS approved for registration as FDIS 2019-02-19; Pending FDIS ballot based on WG 11 N 18277 as of 2019-12
* ISO/IEC 23008-2:201x (Edition 4)/DAmd 1:201x (Additional supplemental enhancement information); DAM ballot started 2019-07-10, closed 2019-10-02

## On the general decoding process

### Status

This item has been confirmed by the JCT-VC, and the proposed fix has been agreed in spirit. For background, see [JCTVC-AI0022](http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=10923) and [JCTVC-AJ0021](http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=10939). However, the specific text for the fix is not yet available and hence this has not yet been resolved.

### Description of the issue

The general decoding process is specified in a CVS-by-CVS manner, because the sub-bitstream extraction process relies on the SPS syntax element sps\_max\_sub\_layers\_minus1, while the active SPS can change across CVSs.

For each CVS of the input bitstream, the sub-bitstream extraction process is applied first with the output being BitstreamToDecode. Therefore, BitstreamToDecode contains one CVS. After that, the decoding process for a coded picture specified in clause 8.1.3 is repeatedly invoked for each coded picture in BitstreamToDecode.

The following is stated in clause 8.1.3:

*The decoding process for the current picture takes as inputs the syntax elements and upper-case variables from clause 7. When interpreting the semantics of each syntax element in each NAL unit, the term "the bitstream" (or part thereof, e.g., a CVS of the bitstream) refers to BitstreamToDecode (or part thereof).*

From the yellow-highlighted wording, it is clear that the term "the bitstream" mentioned above was intended to mean something that can contain more than one CVS.

Furthermore, there are many places related to "*interpreting the semantics of each syntax element in each NAL unit*" that have the wording "the first picture in the bitstream", particularly when used to determine whether a CRA picture starts a CVS. In my understanding, in those places, the intent was that the term "the bitstream" mentioned above was intended to mean something that can contain more than one CVS.

Lastly, at the beginning of the general decoding process for the input bitstream, strictly speaking, determining the CVSs in the input bitstream would not be clear before determination of the variable NoRaslOutputFlag for each IRAP picture, which currently is only part of the decoding process of a CVS.

### Description of the proposed fix

It is proposed to change the description of the general decoding process as follows:

1. Firstly the CVSs in the input bitstream are identified.
2. Sub-bitstream extraction is applied for each CVS.
3. The extracted results of all the CVSs are concatenated to BitstreamToDecode.
4. BitstreamToDecode is decoded CVS by CVS, and each CVS picture by picture.

## On HRD for splicing

### Status

This item has been confirmed by the JCT-VC and resolved for the ITU-T text, but has not yet been resolved for the ISO/IEC text. For background, see [JCTVC-AK0027](http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=10967).

### Description of the issue

It is reported that the current HEVC specification determines, when concatenationFlag is set to 1, that the AuNominalRemovalTime for the splicing point under some circumstances (non-seamless splicing) may be greater than AuNominalRemovalTime[ prevNonDiscardablePic ] + ( auCpbRemovalDelayDeltaMinus1 + 1 ).

The text in HEVC specifies the following:

if( !concatenationFlag ) {  
 baseTime = AuNominalRemovalTime[ firstPicInPrevBuffPeriod ]  
 tmpCpbRemovalDelay = AuCpbRemovalDelayVal  
} else {  
 baseTime = **AuNominalRemovalTime[ prevNonDiscardablePic ]**  
 tmpCpbRemovalDelay =  
 Max( ( auCpbRemovalDelayDeltaMinus1 + 1 ), (C‑10)  
 Ceil( ( **InitCpbRemovalDelay[ SchedSelIdx ] ÷ 90000 +  
 AuFinalArrivalTime[ n − 1 ] − AuNominalRemovalTime[ n − 1 ]** ) ÷ ClockTick ) )  
}  
AuNominalRemovalTime[ n ] = baseTime + ClockTick \* ( tmpCpbRemovalDelay − CpbDelayOffset )

In the following figure, a splicing operation is shown that reportedly does not allow for seamless splicing since the InitCpbRemovalDelay[ SchedSelIdx ] imposes a later removal time after splicing than equidistant CPB removal times as present in seamless playout.



If instead of deriving the removal time of AU 6 from the AU n-1 (i.e. AU 5) as indicated above in the figure, we use the last non-discardable picture (AU 4), this would reportedly result in the following:

taf6= trm4 + >= trm6 (CPB underflow resulting in non-seamless splicing)

trm6= taf5+InitCpbRemovalDelay

=taf6 – trm4= Size / Bitrate – ( trm4 – taf5 )

trm4 + Size / Bitrate – ( trm4 – taf5 ) >= taf5 + InitCpbRemovalDelay

Size / Bitrate – ( trm4 – taf5 ) >= ( taf5 – trm4 )+ InitCpbRemovalDelay

taf5 + Size / Bitrate = **taf6 >= trm4 + ( taf5 – trm4 ) + InitCpbRemovalDelay**

In the derived formula, the part marked in yellow highlight and boldface in Equation C-10 above is reportedly not correct.

### Description of the proposed fix

The proposed fix is to change Equation C-10 to the following:

if( !concatenationFlag ) {  
 baseTime = AuNominalRemovalTime[ firstPicInPrevBuffPeriod ]  
 tmpCpbRemovalDelay = AuCpbRemovalDelayVal  
} else {  
 baseTime1 = AuNominalRemovalTime[ prevNonDiscardablePic ]  
 tmpCpbRemovalDelay1 = ( auCpbRemovalDelayDeltaMinus1 + 1 )  
 baseTime2 = AuNominalRemovalTime[ n − 1 ]  
 tmpCpbRemovalDelay2 =   
 Ceil( ( InitCpbRemovalDelay[ SchedSelIdx ] ÷ 90000 +  
 AuFinalArrivalTime[ n − 1 ] − AuNominalRemovalTime[ n − 1 ] ) ÷ ClockTick ) (C‑10)  
 if( baseTime1 + ClockTick \* tmpCpbRemovalDelay1 <   
 baseTime2 + ClockTick \* tmpCpbRemovalDelay2 ) {  
 baseTime = baseTime2  
 tmpCpbRemovalDelay = tmpCpbRemovalDelay2  
 } else {  
 baseTime = baseTime1  
 tmpCpbRemovalDelay = tmpCpbRemovalDelay1  
 }  
}  
AuNominalRemovalTime[ n ] = baseTime + ClockTick \* ( tmpCpbRemovalDelay − CpbDelayOffset )

## On filtering process for chroma block edges

### Status

This item has been confirmed by the JCT-VC and resolved for the ITU-T text, but has not yet been resolved for the ISO/IEC text.

### Bug fix

*In 8.7.2.5.5 (Filtering process for chroma block edges), change the following:*

If ChromaArrayType is equal to 1, the variable QpC is determined as specified in Table ‎8‑10 based on the index qPi derived as follows:

qPi = ( ( QpQ + QpP + 1 )  >>  1 ) + cQpPicOffset (‎8‑384)

Otherwise (ChromaArrayType is greater than 1), the variable QpC is set equal to Min( qPi, 51 ).

*to*

The index qPi derived as follows:

qPi = ( ( QpQ + QpP + 1 )  >>  1 ) + cQpPicOffset (8‑384)

The variable QpC is derived as follows:

– If ChromaArrayType is equal to 1, the variable QpC is determined based on qPi as specified in Table 8‑10.

– Otherwise (ChromaArrayType is greater than 1), the variable QpC is set equal to Min( qPi, 51 ).

## On semantics of nal\_hrd\_parameters\_present\_flag and vcl\_hrd\_parameters\_present\_flag

### Status

This item has been confirmed by the JCT-VC and resolved for the ITU-T text, but has not yet been resolved for the ISO/IEC text.

### Bug fixes

1. *In E.3.2 (HRD parameters semantics), change the following:*

**nal\_hrd\_parameters\_present\_flag** equal to 1 specifies that NAL HRD parameters (pertaining to Type II bitstream conformance) are present in the hrd\_parameters( ) syntax structure. nal\_hrd\_parameters\_present\_flag equal to 0 specifies that NAL HRD parameters are not present in the hrd\_parameters( ) syntax structure.

NOTE 1 – When nal\_hrd\_parameters\_present\_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the NAL HRD parameters and all buffering period and picture timing SEI messages, by some means not specified in this Specification.

*to*

**nal\_hrd\_parameters\_present\_flag** equal to 1 specifies that NAL HRD parameters (pertaining to the Type II bitstream conformance point) are present in the hrd\_parameters( ) syntax structure. nal\_hrd\_parameters\_present\_flag equal to 0 specifies that NAL HRD parameters are not present in the hrd\_parameters( ) syntax structure.

NOTE 1 – When nal\_hrd\_parameters\_present\_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the NAL HRD parameters and all buffering period SEI messages, and, when vcl\_hrd\_parameters\_present\_flag is also equal to 0, all picture timing and decoding unit information SEI messages, by some means not specified in this Specification.

1. *In E.3.2 (HRD parameters semantics), change the following:*

**vcl\_hrd\_parameters\_present\_flag** equal to 1 specifies that VCL HRD parameters (pertaining to all bitstream conformance) are present in the hrd\_parameters( ) syntax structure. vcl\_hrd\_parameters\_present\_flag equal to 0 specifies that VCL HRD parameters are not present in the hrd\_parameters( ) syntax structure.

NOTE 2 – When vcl\_hrd\_parameters\_present\_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the VCL HRD parameters and all buffering period and picture timing SEI messages, by some means not specified in this Specification.

*to*

**vcl\_hrd\_parameters\_present\_flag** equal to 1 specifies that VCL HRD parameters (pertaining to the Type I bitstream conformance point) are present in the hrd\_parameters( ) syntax structure. vcl\_hrd\_parameters\_present\_flag equal to 0 specifies that VCL HRD parameters are not present in the hrd\_parameters( ) syntax structure.

NOTE 2 – When vcl\_hrd\_parameters\_present\_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the VCL HRD parameters and all buffering period SEI messages, and, when nal\_hrd\_parameters\_present\_flag is also equal to 0, all picture timing and decoding unit information SEI messages, by some means not specified in this Specification.

## On setting of HighestTid to sps\_max\_sub\_layers\_minus1

### Status

This item has been confirmed by the JCT-VC and resolved for the ITU-T text, but has not yet been resolved for the ISO/IEC text.

### Description of the issue

The phrase of "HighestTid is set equal to sps\_max\_sub\_layers\_minus1" in clauses F.8.1.2 is not clear, as there may be multiple layers referring to multiple SPSs and consequently there may be multiple instances of sps\_max\_sub\_layers\_minus1 involved.

### Bug fix

*In F.8.1.2 (CVSG decoding process) change the following:*

– Otherwise, HighestTid is set equal to sps\_max\_sub\_layers\_minus1.

*to*

– Otherwise, HighestTid is set equal to vps\_max\_sub\_layers\_minus1.

## On HRD text that involves using NoOutputOfPriorPicsFlag instead of no\_output\_of\_prior\_pics\_flag and mentioning decoding unit information SEI messages under some circumstances

### Status

This item has been confirmed by the JCT-VC and resolved for the ITU-T text, but has not yet been resolved for the ISO/IEC text.

### Bug fixes

1. *In C.3.3, change the following:*

Otherwise (PicOutputFlag is equal to 1 and DpbOutputTime[ n ] is greater than AuCpbRemovalTime[ n ] ), the current picture is output later and will be stored in the DPB (as specified in clause ‎C.3.4) and is output at time DpbOutputTime[ n ] unless indicated not to be output by the decoding or inference of no\_output\_of\_prior\_pics\_flag equal to 1 at a time that precedes DpbOutputTime[ n ].

*to*

Otherwise (PicOutputFlag is equal to 1 and DpbOutputTime[ n ] is greater than AuCpbRemovalTime[ n ] ), the current picture is output later and will be stored in the DPB (as specified in clause ‎C.3.4) and is output at time DpbOutputTime[ n ] unless indicated not to be output by NoOutputOfPriorPicsFlag equal to 1.

1. *In F.13.3.3, change the following:*

Otherwise (PicOutputFlag is equal to 1 and DpbOutputTime[ n ] is greater than AuCpbRemovalTime[ n ] ), the current picture is output later and will be stored in the DPB (as specified in clause ‎F.13.3.4) and is output at time DpbOutputTime[ n ] unless indicated not to be output by the decoding or inference of NoOutputOfPriorPicsFlag equal to 1 at a time that precedes DpbOutputTime[ n ].

*to*

Otherwise (PicOutputFlag is equal to 1 and DpbOutputTime[ n ] is greater than AuCpbRemovalTime[ n ] ), the current picture is output later and will be stored in the DPB (as specified in clause ‎F.13.3.4) and is output at time DpbOutputTime[ n ] unless indicated not to be output by NoOutputOfPriorPicsFlag equal to 1.

1. *In C.5.1, change the following:*

A decoder claiming conformance to a specific profile, tier and level shall be able to successfully decode all bitstreams that conform to the bitstream conformance requirements specified in clause ‎C.4, in the manner specified in Annex ‎A, provided that all VPSs, SPSs and PPSs referred to in the VCL NAL units and appropriate buffering period and picture timing SEI messages are conveyed to the decoder, in a timely manner, either in the bitstream (by non-VCL NAL units), or by external means not specified in this Specification.

*to*

A decoder claiming conformance to a specific profile, tier and level shall be able to successfully decode all bitstreams that conform to the bitstream conformance requirements specified in clause ‎C.4, in the manner specified in Annex ‎A, provided that all VPSs, SPSs and PPSs referred to in the VCL NAL units and appropriate buffering period, picture timing, and decoding unit information SEI messages are conveyed to the decoder, in a timely manner, either in the bitstream (by non-VCL NAL units), or by external means not specified in this Specification.

## On whether the pan-scan rectangle SEI message should be in the SingleLayerSeiList

### Status

This item has been confirmed by the JCT-VC and resolved for the ITU-T text, but has not yet been resolved for the ISO/IEC text.

### Description of the issue

In D.3.1, there is a discrepancy (of the SEI payloadType 2, i.e., the pan-scan rectangle SEI message) between the definition of the variable SingleLayerSeiList and the subsequent NOTE providing an explanation of SingleLayerSeiList, copied and pasted below:

The list SingleLayerSeiList is set to consist of the payloadType values 3, 6, 9, 15, 16, 17, 19, 22, 23, 45, 47, 56, 128, 129, 131, 132, 134 to 151, inclusive, 154 to 159, and 200 to 201, inclusive.

NOTE 3 – SingleLayerSeiList consists of the payloadType values of the SEI messages specified in Annex D excluding 0 (buffering period), 1 (picture timing), 4 (user data registered by Recommendation ITU-T T.35), 5 (user data unregistered), 130 (decoding unit information) and 133 (scalable nesting).

Due to that this list is mainly used to constrain the value of bitstream\_subset\_flag of the scalable nesting SEI message, and that the value 2 is the pan-scan rectangle SEI message, I think the NOTE is correct, i.e., the value 2 should be added to the definition. This can be confirmed by checking the editing history, e.g., by taking a look at JCTVC-R1008-v1 of the Sapporo meeting in July 2014.

### Bug fix

*In D.3.1 (General SEI payload semantics), change the following:*

The list SingleLayerSeiList is set to consist of the payloadType values 3, 6, 9, 15, 16, 17, 19, 22, 23, 45, 47, 56, 128, 129

*to*

The list SingleLayerSeiList is set to consist of the payloadType values 2, 3, 6, 9, 15, 16, 17, 19, 22, 23, 45, 47, 56, 128, 129

## On the semantics of rbsp\_byte[ i ]

### Status

These bugs were confirmed, and the text bug fixes were agreed by the JCT-VC at its 37th meeting in Geneva in Oct. 2019. Resolved for the ITU-T text, but not yet for the ISO/IEC text. See [JCTVC-AK0023](http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=10963).

### Bug fixes

*Change the semantics of rbsp\_byte[ i ] as follows:*

**rbsp\_byte**[ i ] is the i-th byte of an RBSP. An RBSP is specified as an ordered sequence of bytes as follows:

The RBSP contains a string of data bits (SODB) as follows:

– If the SODB is empty (i.e., zero bits in length), the RBSP is also empty.

– Otherwise, the RBSP contains the SODB as follows:

1) The first byte of the RBSP contains the first (most significant, left-most) eight bits of the SODB; the next byte of the RBSP contains the next eight bits of the SODB, etc., until fewer than eight bits of the SODB remain.

2) The rbsp\_trailing\_bits( ) syntax structure is present after the SODB as follows:

i) The first (most significant, left-most) bits of the final RBSP byte contain the remaining bits of the SODB (if any).

ii) The next bit consists of a single bit equal to 1 (i.e., rbsp\_stop\_one\_bit).

iii) When the rbsp\_stop\_one\_bit is not the last bit of a byte-aligned byte, one or more zero-valued bits (i.e., instances of rbsp\_alignment\_zero\_bit) are present to result in byte alignment.

3) One or more cabac\_zero\_word 16-bit syntax elements equal to 0x0000 may be present in some RBSPs after the rbsp\_trailing\_bits( ) at the end of the RBSP.

# Reported errata items for AVC

See also section 1.

## Publication status background

Rec. ITU-T H.264

* (06/19, Edition 13) Approved 2019-06-13, published 2019-09-06

ISO/IEC 14496-10

* ISO/IEC 14496-10:2014 (Edition 8), published 2014-09
* ISO/IEC 14496-10:2014/Amd 1:2015 (Multi-resolution frame compatible stereoscopic video with depth maps, additional supplemental enhancement information and video usability information), published 2015-11
* ISO/IEC 14496-10:2014/FDAMD 2 (Additional Levels and Supplemental Enhancement Information); stage 50.98, deleted in preparation for Edition 9
* ISO/IEC 14496-10:2014/Amd 3:2016 (Additional supplemental enhancement information); published 2016-12, published 2016-12
* ISO/IEC DIS 14496-10:201x (Edition 9); stage 40.99

## Text bug fixes for Annexes I and J

### Status

These bugs were confirmed, and the text bug fixes were agreed by the JCT-VC at its 37th meeting in Geneva in Oct. 2019. See Section 1 of [JCTVC-AK0022](http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=10962).

### Bug fixes



*In I.13.2.3.1, change the semantics of da\_mantissa\_len\_minus1 to the following:*

**da\_mantissa\_len\_minus1** + 1 specifies the number of bits in the da\_mantissa syntax element. The value of da\_mantissa\_len\_minus1 shall be in the range of 0 to 31, inclusive. The variable OutManLen is set equal to da\_mantissa\_len\_minus1 + 1.

*In J.7.3.2.13.1, change the depth ranges syntax to the following:*

|  |  |  |
| --- | --- | --- |
| depth\_ranges( numViews, predDirection, index ) { | C | Descriptor |
| **z\_near\_flag** | 11 | u(1) |
| **z\_far\_flag** | 11 | u(1) |
| if( z\_near\_flag ) |  |  |
| 3dv\_acquisition\_element( numViews, predDirection, 7, index, ZNearSign,  ZNearExp, ZNearMantissa, ZNearManLen ) |  |  |
| if( z\_far\_flag ) |  |  |
| 3dv\_acquisition\_element( numViews, predDirection, 7, index, ZFarSign,  ZFarExp, ZFarMantissa, ZFarManLen ) |  |  |
| } |  |  |

*In J.7.3.2.13.2, change the 3DV acquisition element syntax as follows:*

|  |  |  |
| --- | --- | --- |
| 3dv\_acquisition\_element( numViews, predDirection, expLen, index, outSign, outExp, outMantissa, outManLen ) { | **C** | Descriptor |
| if( numViews > 1 ) |  |  |
| **element\_equal\_flag** | 11 | u(1) |
| if( element\_equal\_flag  = =  0 ) |  |  |
| numValues = numViews |  |  |
| else |  |  |
| numValues = 1 |  |  |
| for( i = 0; i < numValues; i++ ) { |  |  |
| if( predDirection  = =  2  &&  i  = =  0 ) { |  |  |
| **mantissa\_len\_minus1** | 11 | u(5) |
| outManLen[ index, i ] = manLen = mantissa\_len\_minus1 + 1 |  |  |
| } |  |  |
| if( predDirection  = =  2 ) { |  |  |
| **sign0** | 11 | u(1) |
| outSign[ index, i ] = sign0 |  |  |
| **exponent0** | 11 | u(v) |
| outExp[ index, i ] = exponent0 |  |  |
| **mantissa0** | 11 | u(v) |
| outMantissa[ index, i ] = mantissa0 |  |  |
| } else { |  |  |
| **skip\_flag** | 11 | u(1) |
| if( skip\_flag = = 0 ) { |  |  |
| **sign1** | 11 | u(1) |
| outSign[ index, i ] = sign1 |  |  |
| **exponent\_skip\_flag** | 11 | u(1) |
| if( exponent\_skip\_flag = = 0 ) { |  |  |
| **exponent1** | 11 | u(v) |
| outExp[ index, i ] = exponent1 |  |  |
| } else |  |  |
| outExp[ index, i ] = outExp[ ref\_dps\_id0, i ] |  |  |
| **mantissa\_diff** | 11 | se(v) |
| if( predDirection = = 0 ) |  |  |
| mantissaPred = (( outMantissa[ ref\_dps\_id0, i ] \* predWeight0 +  outMantissa[ ref\_dps\_id1, i ] \* ( 64-predWeight0 ) + 32 ) >> 6 ) |  |  |
| else |  |  |
| mantissaPred = outMantissa[ ref\_dps\_id0, i ] |  |  |
| outMantissa[ index, i ] = mantissaPred + mantissa\_diff |  |  |
| outManLen[ index, i ] = outManLen[ ref\_dps\_id0, i ] |  |  |
| } else { |  |  |
| outSign[ index, i ] = outSign[ ref\_dps\_id0, i ] |  |  |
| outExp[ index, i ] = outExp[ ref\_dps\_id0, i ] |  |  |
| outMantissa[ index, i ] = outMantissa[ ref\_dps\_id0, i ] |  |  |
| outManLen[ index, i ] = outManLen[ ref\_dps\_id0, i ] |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| if( element\_equal\_flag = = 1 ) { |  |  |
| for( i = 1; i < numViews; i++ ) { |  |  |
| outSign[ index, i ] = outSign[ index, 0 ] |  |  |
| outExp[ index, i ] = outExp[ index, 0 ] |  |  |
| outMantissa[ index, i ] = outMantissa[ index, 0 ] |  |  |
| outManLen[ index, i ] = outManLen[ index, 0 ] |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

*In J.7.4.2.13.2, change the 3DV acquisition element semantics as follows:*

The syntax structure specifies the value of an element in the depth ranges syntax structure. The element may contain one or more loop entries i of the order specified by view\_id\_3dv syntax elements.

The contents of the syntax structure are controlled through input variables predDirection, expLen, and index, the semantics of which are as follows.

– predDirection equal to 2 specifies that the first loop entry of the element is not predicted and coded in the sign, exponent, and mantissa syntax elements. predDirection equal to 0 or 1 specifies that the first loop entry of the element is predicted and a difference relative to a prediction value is coded in the difference syntax element.

– expLen specifies the number of bits in the exponent syntax element.

– index greater than 0 specifices the depth\_parameter\_set\_id of the depth parameter set wherein the parameters are present, and index equal to 0 specifies that the parameters are present in a sequence parameter set.

The syntax structure uses outSign, outExp, outMantissa, and outManLen variables for both input and output, where each variable is indexed by [ index, viewIdc ], index being an identifier (equal to either 0 when decoding depth ranges in sequence parameter set or depth\_parameter\_set\_id value when decoding depth range parameter set) to a depth parameter set and viewIdc being a view indicator (in the order of views for 3DV acquisition parameters).

**element\_equal\_flag** equal to 0 specifies that the sign, exponent, and mantissa may or may not be identical to respective values for any two loop entries i and j. element\_equal\_flag equal to 1 specifies that the sign, exponent, and mantissa are identical to respective values for any two loop entries i and j. When not present, element\_equal\_flag is inferred to be equal to 0.

**mantissa\_len\_minus1** plus 1 specifies the number of bits in the mantissa syntax element. The value of mantissa\_len\_minus1 shall be in the range of 0 to 31, inclusive.

**sign0** equal to 0 indicates that the sign of the value provided in the loop entry is positive. sign0 equal to 1 indicates that the sign is negative.

**exponent0** specifies the exponent of the value provided by the loop entry. The syntax element exponent0 is represented by expLen bits. The value of exponent0 shall be in the range of 0 to 2expLen – 2, inclusive. The value 2expLen – 1 is reserved for future use by ITU‑T | ISO/IEC. Decoders shall treat the value 2expLen – 1 as indicating an unspecified value.

**mantissa0** specifies the mantissa of the value provided by the loop entry. The syntax element mantissa0 is represented by manLen bits.

**skip\_flag** equal to 0 specifies that syntax elements sign1, exponent\_skip\_flag and mantissa\_diff are present for the loop entry. skip\_flag equal to 1 specifies that elements sign1, exponent\_skip\_flag and mantissa\_diff are not present for the loop entry.

**sign1** equal to 0 indicates that the sign of the value provided in the loop entry is positive. sign1 equal to 1 indicates that the sign is negative.

**exponent1**, if present, specifies the exponent of the value provided by the loop entry. The syntax element exponent1 is represented by expLen bits. The value of exponent1 shall be in the range of 0 to 2expLen – 2, inclusive. The value 2expLen – 1 is reserved for future use by ITU‑T | ISO/IEC. Decoders shall treat the value 2expLen – 1 as indicating an unspecified value.

**mantissa\_diff** specifies the difference of the mantissa of the value provided by the loop entry relative to its prediction value.

## On semantics of nal\_hrd\_parameters\_present\_flag and vcl\_hrd\_parameters\_present\_flag

### Status

These bugs were confirmed, and the text bug fixes were agreed by the JCT-VC at its 37th meeting in Geneva in Oct. 2019. See Section 2 of [JCTVC-AK0022](http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=10962).

### Bug fixes

*Change the semantics of nal\_hrd\_parameters\_present\_flag and vcl\_hrd\_parameters\_present\_flag as follows (additions are yellow-highlighted):*

**nal\_hrd\_parameters\_present\_flag** equal to 1 specifies that NAL HRD parameters (pertaining to the Type II bitstream conformance point) are present. nal\_hrd\_parameters\_present\_flag equal to 0 specifies that NAL HRD parameters are not present.

NOTE 12 – When nal\_hrd\_parameters\_present\_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the NAL HRD parameters and all buffering period SEI messages, and, when vcl\_hrd\_parameters\_present\_flag is also equal to 0, all picture timing SEI messages, by some means not specified in this Recommendation | International Standard.

...

**vcl\_hrd\_parameters\_present\_flag** equal to 1 specifies that VCL HRD parameters (pertaining to the Type I ~~all~~ bitstream conformance point) are present. vcl\_hrd\_parameters\_present\_flag equal to 0 specifies that VCL HRD parameters are not present.

NOTE 13 – When vcl\_hrd\_parameters\_present\_flag is equal to 0, the conformance of the bitstream cannot be verified without provision of the VCL HRD parameters and all buffering period SEI messages, and, when nal\_hrd\_parameters\_present\_flag is also equal to 0, all picture timing SEI messages, by some means not specified in this Recommendation | International Standard.

...

## On semantics of rbsp\_byte[ i ]

### Status

These bugs were confirmed, and the text bug fixes were agreed by the JCT-VC at its 37th meeting in Geneva in Oct. 2019. See [JCTVC-AK0023](http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=10963).

### Bug fixes

*Change the semantics of rbsp\_byte[ i ] to the following:*

**rbsp\_byte**[ i ] is the i-th byte of an RBSP. An RBSP is specified as an ordered sequence of bytes as follows:

The RBSP contains a string of data bits (SODB) as follows:

– If the SODB is empty (i.e., zero bits in length), the RBSP is also empty.

– Otherwise, the RBSP contains the SODB as follows:

1) The first byte of the RBSP contains the first (most significant, left-most) eight bits of the SODB; the next byte of the RBSP contains the next eight bits of the SODB, etc., until fewer than eight bits of the SODB remain.

2) The rbsp\_trailing\_bits( ) syntax structure is present after the SODB as follows:

i) The first (most significant, left-most) bits of the final RBSP byte contain the remaining bits of the SODB (if any).

ii) The next bit consists of a single bit equal to 1 (i.e., rbsp\_stop\_one\_bit).

iii) When the rbsp\_stop\_one\_bit is not the last bit of a byte-aligned byte, one or more zero-valued bits (i.e., instances of rbsp\_alignment\_zero\_bit) are present to result in byte alignment.

3) One or more cabac\_zero\_word 16-bit syntax elements equal to 0x0000 may be present in some RBSPs after the rbsp\_trailing\_bits( ) at the end of the RBSP.

# Reported errata items for Video CICP

See section 1.

## Publication status background

Rec. ITU-T H.273

* (02/16, Edition 1) Approved 2016-12-22, published 2017-04-27

ISO/IEC 23091-2 (previously part of ISO/IEC 23001-8)

* ISO/IEC 23091-2:2019, published 2019-07