**COMMITTEE DRAFT AMENDMENT****© ISO/IEC 2022 – All rights reserved****Text of ISO/IEC 23008-2:2020/CDAM 2** **63****Part 2: High efficiency video coding, AMENDMENT 2: High-range levels****Information technology — High efficiency coding and media delivery in heterogeneous environments****Élément introductif — Élément central — Partie 15: Titre de la partie****Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding, AMENDMENT 2: High-range levels****E****2022-02-18****(30) Committee****ISO/IEC****ISO/IEC J****2020****2****Amendment****International Standard****202x****102****ISO/IEC 23008‑****ISO/IEC 23008‑2****ISO/IEC 23008-2:2020/CDAM 2****JISC****Coding of audio, picture, multimedia and hypermedia information****Information technology****5****29****1** **2****見出し 2****見出し 1****0****2****STD Version 2.1c2****30** **4** **ISO/IEC JTC 1/SC 29 /WG 5 N 102**

Date: **2022-02-18**

**Text of ISO/IEC 23008-2:2020/CDAM 2**

ISO/IEC JTC 1/SC 29/WG 5

Secretariat:  JISC

**Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding, AMENDMENT 2: High-range levels**

*Élément introductif — Élément central — Partie 3: Titre de la partie*

|  |
| --- |
| **Warning**  This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.  Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation |

**Copyright notice**

This ISO document is a working draft or committee draft and is copyright-protected by ISO. While the reproduction of working drafts or committee drafts in any form for use by participants in the ISO standards development process is permitted without prior permission from ISO, neither this document nor any extract from it may be reproduced, stored or transmitted in any form for any other purpose without prior written permission from ISO.

Requests for permission to reproduce this document for the purpose of selling it should be addressed as shown below or to ISO's member body in the country of the requester:

[Indicate the full address, telephone number, fax number, telex number, and electronic mail address, as appropriate, of the Copyright Manger of the ISO member body responsible for the secretariat of the TC or SC within the framework of which the working document has been prepared.]

Reproduction for sales purposes may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see [www.iso.org/directives](https://www.iso.org/directives-and-policies.html)).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see [www.iso.org/patents](https://www.iso.org/iso-standards-and-patents.html)) or the IEC list of patent declarations received (see [http://patents.iec.ch](http://patents.iec.ch/)).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see [www.iso.org/iso/foreword.html](https://www.iso.org/foreword-supplementary-information.html).

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*, in collaboration with ITU-T. Technically aligned twin text is published as Rec. ITU-T H.265.

A list of all parts in the ISO/IEC 23008 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at [www.iso.org/members.html](https://www.iso.org/members.html).

Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding, AMENDMENT 2: High-range levels

*In subclause 5.7, make the following changes:*

x = y..z x takes on integer values starting from y to z, inclusive, with x, y, and z being integer numbers and z being greater than or equal to y.

*In subclause 7.4.2.4.3, make the following changes:*

...

It is a requirement of bitstream conformance that, when present, the next access unit after an access unit that contains an end of sequence NAL unit ~~or an end of bitstream NAL unit~~ shall be an IRAP access unit, which may be an IDR access unit, a BLA access unit, or a CRA access unit.

...

*In subclause 7.4.3.1, make the following changes:*

...

**vps\_extension\_flag** equal to 0 specifies that no vps\_extension\_data\_flag syntax elements are present in the VPS RBSP syntax structure. vps\_extension\_flag equal to 1 specifies that there ~~are~~ may be vps\_extension\_data\_flag syntax elements present in the VPS RBSP syntax structure. Decoders conforming to a profile specified in Annex A but not supporting the INBLD capability specified in Annex F shall ignore all data that follow the value 1 for vps\_extension\_flag in a VPS NAL unit.

...

*In subclause 7.4.3.3.1, make the following changes:*

...

**pps\_deblocking\_filter\_disabled\_flag** equal to 1 specifies that ~~the operation of deblocking filter is not applied for slices referring to the PPS in which slice\_deblocking\_filter\_disabled\_flag is not present~~ the deblocking filter is disabled for pictures referring to the PPS unless overridden for a slice by information present in the slice header. pps\_deblocking\_filter\_disabled\_flag equal to 0 specifies that ~~the operation of the deblocking filter is applied for slices referring to the PPS in which slice\_deblocking\_filter\_disabled\_flag is not present~~ the deblocking filter is enabled for pictures referring to the PPS unless overridden for a slice by information present in the slice header. When not present, the value of pps\_deblocking\_filter\_disabled\_flag is inferred to be equal to 0.

...

*In subclause 8.3.3.1, make the following changes:*

...

When this process is invoked, the following applies:

* For each RefPicSetStFoll[ i ], with i in the range of 0 to NumPocStFoll − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in clause 8.3.3.2, and the following applies:
* The value of PicOrderCntVal for the generated picture is set equal to PocStFoll[ i ].
* The value of PicOutputFlag for the generated picture is set equal to 0.
* The generated picture is marked as "used for short-term reference".
* RefPicSetStFoll[ i ] is set to be the generated reference picture.
* The value of nuh\_layer\_id for the generated picture is set equal to nuh\_layer\_id of the current picture.
* The value of TemporalId for the generated picture is set equal to TemporalId of the current picture.
* The value of slice\_pic\_parameter\_set\_id for the generated picture is set equal to slice\_pic\_parameter\_set\_id of the current picture.
* For each RefPicSetLtFoll[ i ], with i in the range of 0 to NumPocLtFoll − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in clause 8.3.3.2, and the following applies:
* The value of PicOrderCntVal for the generated picture is set equal to PocLtFoll[ i ].
* The value of slice\_pic\_order\_cnt\_lsb for the generated picture is inferred to be equal to ( PocLtFoll[ i ] & ( MaxPicOrderCntLsb − 1 ) ).
* The value of PicOutputFlag for the generated picture is set equal to 0.
* The generated picture is marked as "used for long-term reference".
* RefPicSetLtFoll[ i ] is set to be the generated reference picture.
* The value of nuh\_layer\_id for the generated picture is set equal to nuh\_layer\_id of the current picture.
* The value of TemporalId for the generated picture is set equal to TemporalId of the current picture.
* The value of slice\_pic\_parameter\_set\_id for the generated picture is set equal to slice\_pic\_parameter\_set\_id of the current picture.

...

*In subclause 8.5.3.1, make the following changes:*

...

The decoding process for prediction units in inter prediction mode consists of the following ordered steps:

1. The derivation process for motion vector components and reference indices as specified in clause 8.5.3.2 is invoked with the luma coding block location ( xCb, yCb ), the luma prediction block location ( xBl, yBl ), the luma coding block size block nCbS, the luma prediction block width nPbW, the luma prediction block height nPbH and the prediction unit index partIdx as inputs, and the luma motion vectors mvL0 and mvL1, when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1, the reference indices refIdxL0 and refIdxL1 and the prediction list utilization flags predFlagL0 and predFlagL1 as outputs.
2. The decoding process for inter sample prediction as specified in clause 8.5.3.3 is invoked with the luma coding block location ( xCb, yCb ), the luma prediction block location ( xBl, yBl ), the luma coding block size block nCbS, the luma prediction block width nPbW, the luma prediction block height nPbH, the luma motion vectors mvL0 and mvL1, when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1, the reference indices refIdxL0 and refIdxL1, and the prediction list utilization flags predFlagL0 and predFlagL1 as inputs, and the inter prediction samples (predSamples) that are an (nCbSL)x(nCbSL) array predSamplesL of prediction luma samples and, when ChromaArrayType is not equal to 0, two (nCbSwC)x(nCbShC) arrays ~~predSamples~~~~Cr~~predSamplesCb and predSamplesCr of prediction chroma samples, one for each of the chroma components Cb and Cr, as outputs.

...

*In subclause 8.5.3.2.7, make the following changes, and renumber the equation indices accordingly:*

...

3. The following applies for ( xNbBk, yNbBk ) from ( xNbB0, yNbB0 ) to ( xNbB2, yNbB2 ):

— The availability derivation process for a prediction block as specified in 6.4.2 is invoked with the luma location ( xCb, yCb ), the current luma coding block size nCbS, the luma prediction block location ( xPb, yPb ), the luma prediction block width nPbW, the luma prediction block height nPbH, the luma location ( xNbY, yNbY ) set equal to ( xNbBk, yNbBk ), and the partition index partIdx as inputs, and the output is assigned to the prediction block availability flag availableBk.

— When availableBk is equal to TRUE and availableFlagLXB is equal to 0, the following applies:

— If PredFlagLX[ xNbBk ][ yNbBk ] is equal to 1, and DiffPicOrderCnt( RefPicListX[ RefIdxLX[ xNbBk ][ yNbBk ] ], RefPicListX[ refIdxLX ] ) is equal to 0, availableFlagLXB is set equal to 1 and the following assignments are made:

mvLXB = MvLX[ xNbBk ][ yNbBk ] (8‑184)

~~refIdxB = RefIdxLX[ xNbB~~~~k~~~~][ yNbB~~~~k~~~~] (8‑185)~~

— Otherwise, when PredFlagLY[ xNbBk ][ yNbBk ] (with Y = !X) is equal to 1 and DiffPicOrderCnt( RefPicListY[ RefIdxLY[ xNbBk ][ yNbBk ] ], RefPicListX[ refIdxLX ] ) is equal to 0, availableFlagLXB is set equal to 1 and the following assignments are made:

mvLXB = MvLY[ xNbBk ][ yNbBk ] (8‑186)

~~refIdxB = RefIdxLY[ xNbB~~~~k~~~~][ yNbB~~~~k~~~~] (8‑187)~~

...

*In subclause 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 ).

*In subclause 9.3.1, make the following changes:*

...

In case the request for a value of a syntax element is processed for the syntax element pcm\_flag and the decoded value of pcm\_flag is equal to 1, the decoding engine is initialized after the decoding of any pcm\_alignment\_zero\_bit and all pcm\_sample\_luma and pcm\_sample\_chroma data as specified in clause 9.3.2.6.

The storage process for context variables, Rice parameter initialization states, and palette predictor variables is applied as follows:

* + When ending the parsing of the CTU syntax in clause 7.3.8.2, entropy\_coding\_sync\_enabled\_flag is equal to 1 and either CtbAddrInRs % PicWidthInCtbsY is equal to 1 or both CtbAddrInRs is greater than 1 and TileId[ CtbAddrInTs ] is not equal to TileId[ CtbAddrRsToTs[ CtbAddrInRs − 2 ] ], the storage process for context variables, Rice parameter initialization states, and palette predictor variables as specified in clause 9.3.2.4 is invoked with TableStateIdxWpp, TableMpsValWpp, TableStatCoeffWpp when persistent\_rice\_adaptation\_enabled\_flag is equal to 1, and PredictorPaletteSizeWpp and PredictorPaletteEntriesWpp when palette\_mode\_enabled\_flag is equal to 1 as outputs.
  + When ending the parsing of the general slice segment data syntax in clause 7.3.8.1, dependent\_slice\_segments\_enabled\_flag is equal to 1 and end\_of\_slice\_segment\_flag is equal to 1, the storage process for context variables, Rice parameter initialization states, and palette predictor variables as specified in clause 9.3.2.4 is invoked with TableStateIdxDs, TableMpsValDs, TableStatCoeffDs when persistent\_rice\_adaptation\_enabled\_flag is equal to 1, and PredictorPaletteSizeDs and PredictorPaletteEntriesDs when palette\_mode\_enabled\_flag is equal to 1 as outputs.

...

*In subclause 9.3.2.1, make the following changes:*

...

— Otherwise, ~~if~~ when CtbAddrInRs is equal to slice\_segment\_address and dependent\_slice\_segment\_flag is equal to 1, the synchronization process for context variables and Rice parameter initialization states as specified in 9.3.2.5 is invoked with TableStateIdxDs, TableMpsValDs, TableStatCoeffDs, PredictorPaletteSizeDs, and TablePredictorPaletteEntriesDs as inputs.

~~— Otherwise, the following applies:~~

~~— The initialization process for context variables is invoked as specified in 9.3.2.2.~~

~~— The variables StatCoeff[ k ] are set equal to 0, for k in the range 0 to 3, inclusive.~~

~~— The initialization process for palette predictor variables is invoked as specified in 9.3.2.3.~~

The initialization process for the arithmetic decoding engine is invoked as specified in 9.3.2.6.

...

*In subclause 9.3.2.4, replace the following:*

Inputs to this process are

— the CABAC context variables indexed by ctxTable and ctxIdx,

— the Rice parameter initialization states indexed by k, and

— the palette predictor variables, PredictorPaletteSize and PredictorPaletteEntries.

Outputs of this process are

— the variables tableStateSync and tableMPSSync containing the values of the variables pStateIdx and valMps used in the initialization process of context variables and Rice parameter initialization states that are assigned to all syntax elements in 7.3.8.1 through 7.3.8.12, except end\_of\_slice\_segment\_flag, end\_of\_subset\_one\_bit, and pcm\_flag,

— the variables tableStatCoeffSync containing the values of the variables StatCoeff[ k ] used in the initialization process of context variables and Rice parameter initialization states, and

— the variables PredictorPaletteSizeSync and tablePredictorPaletteEntriesSync containing the values used in the initialization process of palette predictor variables.

For each context variable, the corresponding entries pStateIdx and valMps of tables tableStateSync and tableMPSSync are initialized to the corresponding pStateIdx and valMps.

For each Rice parameter initialization state k, each entry of the table tableStatCoeffSync is initialized to the corresponding value of StatCoeff[ k ].

For palette predictor variables, PredictorPaletteSizeSync is initialized to PredictorPaletteSize. For tablePredictorPaletteEntriesSync, each entry is initialized to the corresponding value of PredictorPaletteEntries.

*with the following:*

Inputs to this process are:

– The CABAC context variables indexed by ctxTable and ctxIdx.

– The Rice parameter initialization states indexed by k.

– The palette predictor variables, PredictorPaletteSize and PredictorPaletteEntries.

Outputs of this process are:

– The arrays TableStateIdxWpp and TableMpsValWpp containing the values of the variables pStateIdx and valMps used in the initialization process of context variables and Rice parameter initialization states that are assigned to all syntax elements in clauses 7.3.8.1 through 7.3.8.12, except end\_of\_slice\_segment\_flag, end\_of\_subset\_one\_bit and pcm\_flag.

– The array TableStatCoeffWpp containing the values of the variables StatCoeff[ k ] used in the initialization process of context variables and Rice parameter initialization states.

– The arrays PredictorPaletteSizeWpp and PredictorPaletteEntriesWpp containing the values used in the initialization process of palette predictor variables.

For each context variable, the values of pStateIdx and valMps are stored into the corresponding entries in the arrays TableStateIdx0Wpp and TableStateIdx1Wpp, respectively.

For each Rice parameter initialization state k, the value of StatCoeff[ k ] is stored into the corresponding entry in the array TableStatCoeffWpp.

For each palette predictor, the value of PredictorPaletteSize is stored into the corresponding entry in the array PredictorPaletteSizeWpp, and the value of PredictorPaletteEntries is stored into the corresponding entry in the array PredictorPaletteEntriesWpp.

*In subclause 9.3.2.5, replace the content of the subclasue with the following:*

Inputs to this process are:

– The arrays TableStateIdxWpp and TableMpsValWpp containing the values of the variables pStateIdx and valMps used in the storage process of context variables that are assigned to all syntax elements in clauses 7.3.8.1 through 7.3.8.12, except end\_of\_slice\_segment\_flag, end\_of\_subset\_one\_bit and pcm\_flag.

– The array TableStatCoeffWpp containing the values of the variables StatCoeff[ k ] used in the storage process of context variables and Rice parameter initialization states.

– The arrays PredictorPaletteSizeWpp and PredictorPaletteEntriesWpp containing the values used in the storage process of palette predictor variables.

Outputs of this process are:

– The initialized CABAC context variables indexed by ctxTable and ctxIdx.

– The initialized Rice parameter initialization states StatCoeff indexed by k.

– The palette predictor variables, PredictorPaletteSize and PredictorPaletteEntries.

For each context variable, the values of the variables pStateIdx and valMps are set equal to the the corresponding entries in the arrays TableStateIdxWpp and TableMpsValWpp, respectively.

For each Rice parameter initialization state, the variable StatCoeff[ k ] is set equal to the corresponding entry in the array TableStatCoeffWpp.

For each palette predictor, the variable PredictorPaletteSize is set equal to the corresponding entry in the array PredictorPaletteSizeWpp, and the variable PredictorPaletteEntries is set equal to the corresponding entry in the array PredictorPaletteEntriesWpp.

*In subclause 9.3.4.1, replace the following:*

The parsing of each bin is specified by the following two ordered steps:

1. The derivation process for ctxTable, ctxIdx, and bypassFlag as specified in 9.3.4.2 is invoked with binIdx as input and ctxTable, ctxIdx, and bypassFlag as outputs.

2. The arithmetic decoding process as specified in 9.3.4.3 is invoked with ctxTable, ctxIdx, and bypassFlag as inputs and the value of the bin as output.

*with the following:*

The parsing of each bin is performed by invoking the derivation process for ctxTable, ctxIdx, and bypassFlag as specified in clause 9.3.4.2 with binIdx as input and ctxTable, ctxIdx and bypassFlag as outputs.

NOTE – As a consequence of invoking the process specified in clause 9.3.4.2, the arithmetic decoding process as specified in clause 9.3.4.3 is invoked with ctxTable, ctxIdx and bypassFlag as inputs and the value of the bin as output.

*In subclause 9.3.4.2.1, remove the last four rows of the table (i.e., for syntax elements cu\_qp\_delta\_abs, cu\_qp\_delta\_sign\_flag, cu\_chroma\_qp\_offset\_flag, andcu\_chroma\_qp\_offset\_idx. These rows are duplicates of rows appearing earlier in the same table.*

*In subclause A.4.1, replace Table A.8 with the following:*

**Table A.8 – General tier and level limits**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Level** | **Max luma picture size MaxLumaPs (samples)** | **Max CPB size MaxCPB (CpbVclFactor or CpbNalFactor bits)** | | **Max slice segments per picture MaxSliceSegmentsPerPicture** | **Max # of tile rows MaxTileRows** | **Max # of tile columns MaxTileCols** |
| **Main tier** | **High tier** |
| **1** | 36 864 | 350 | - | 16 | 1 | 1 |
| **2** | 122 880 | 1 500 | - | 16 | 1 | 1 |
| **2.1** | 245 760 | 3 000 | - | 20 | 1 | 1 |
| **3** | 552 960 | 6 000 | - | 30 | 2 | 2 |
| **3.1** | 983 040 | 10 000 | - | 40 | 3 | 3 |
| **4** | 2 228 224 | 12 000 | 30 000 | 75 | 5 | 5 |
| **4.1** | 2 228 224 | 20 000 | 50 000 | 75 | 5 | 5 |
| **5** | 8 912 896 | 25 000 | 100 000 | 200 | 11 | 10 |
| **5.1** | 8 912 896 | 40 000 | 160 000 | 200 | 11 | 10 |
| **5.2** | 8 912 896 | 60 000 | 240 000 | 200 | 11 | 10 |
| **6** | 35 651 584 | 60 000 | 240 000 | 600 | 22 | 20 |
| **6.1** | 35 651 584 | 120 000 | 480 000 | 600 | 22 | 20 |
| **6.2** | 35 651 584 | 240 000 | 800 000 | 600 | 22 | 20 |
| **6.3** | 80 216 064 | 240 000 | 1 600 000 | 600 | 22 | 20 |
| **7** | 142 606 336 | 240 000 | 1 600 000 | 1 800 | 44 | 40 |
| **7.1** | 142 606 336 | 480 000 | 3 200 000 | 1 800 | 44 | 40 |
| **7.2** | 142 606 336 | 960 000 | 6 400 000 | 1 800 | 44 | 40 |

*In subclause A.4.2, replace Table A.9 with the following:*

**Table A.9 – Tier and level limits for the video profiles**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Level** | **Max luma sample rate MaxLumaSr (samples/sec)** | **Max bit rate MaxBR (BrVclFactor or BrNalFactor bits/s)** | | **Min compression ratio MinCrBase** | | |
| **Main tier** | **High tier** | **Main tier** | **High tier** | |
| **1** | 552 960 | 128 | - | 2 | 2 | |
| **2** | 3 686 400 | 1 500 | - | 2 | 2 |
| **2.1** | 7 372 800 | 3 000 | - | 2 | 2 |
| **3** | 16 588 800 | 6 000 | - | 2 | 2 |
| **3.1** | 33 177 600 | 10 000 | - | 2 | 2 |
| **4** | 66 846 720 | 12 000 | 30 000 | 4 | 4 |
| **4.1** | 133 693 440 | 20 000 | 50 000 | 4 | 4 |
| **5** | 267 386 880 | 25 000 | 100 000 | 6 | 4 |
| **5.1** | 534 773 760 | 40 000 | 160 000 | 8 | 4 |
| **5.2** | 1 069 547 520 | 60 000 | 240 000 | 8 | 4 |
| **6** | 1 069 547 520 | 60 000 | 240 000 | 8 | 4 |
| **6.1** | 2 139 095 040 | 120 000 | 480 000 | 8 | 4 |
| **6.2** | 4 278 190 080 | 240 000 | 800 000 | 6 | 4 |
| **6.3** | 4 812 963 840 | 320 000 | 1 600 000 | 6 | 4 |
| **7** | 4 812 963 840 | 320 000 | 1 600 000 | 6 | 4 |
| **7.1** | 8 556 380 160 | 480 000 | 1 600 000 | 6 | 4 |
| **7.2** | 17 112 760 320 | 960 000 | 3 200 000 | 6 | 4 |

*In subclause A.4.3, replace Tables A.11 and A.12 with the following:*

**Table****A.11 – Maximum picture rates (pictures per second) at levels 1 to 4.1 for some example picture sizes  
when MinCbSizeY is equal to 64**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Level:** |  |  |  | **1** | **2** | **2.1** | **3** | **3.1** | **4** | **4.1** |
| **Max luma picture size (samples):** |  |  |  | 36 864 | 122 880 | 245 760 | 552 960 | 983 040 | 2 228 224 | 2 228 224 |
| **Max luma sample rate (samples/sec)** |  |  |  | 552 960 | 3 686 400 | 7 372 800 | 16 588 800 | 33 177 600 | 66 846 720 | 133 693 440 |
| **Format nickname** | **Luma width** | **Luma height** | **Luma picture size** |  |  |  |  |  |  |  |
| **SQCIF** | **128** | **96** | 16 384 | 33.7 | 225.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **QCIF** | **176** | **144** | 36 864 | 15.0 | 100.0 | 200.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **QVGA** | **320** | **240** | 81 920 | - | 45.0 | 90.0 | 202.5 | 300.0 | 300.0 | 300.0 |
| **525 SIF** | **352** | **240** | 98 304 | - | 37.5 | 75.0 | 168.7 | 300.0 | 300.0 | 300.0 |
| **CIF** | **352** | **288** | 122 880 | - | 30.0 | 60.0 | 135.0 | 270.0 | 300.0 | 300.0 |
| **525 HHR** | **352** | **480** | 196 608 | - | - | 37.5 | 84.3 | 168.7 | 300.0 | 300.0 |
| **625 HHR** | **352** | **576** | 221 184 | - | - | 33.3 | 75.0 | 150.0 | 300.0 | 300.0 |
| **Q720p** | **640** | **360** | 245 760 | - | - | 30.0 | 67.5 | 135.0 | 272.0 | 300.0 |
| **VGA** | **640** | **480** | 327 680 | - | - | - | 50.6 | 101.2 | 204.0 | 300.0 |
| **525 4SIF** | **704** | **480** | 360 448 | - | - | - | 46.0 | 92.0 | 185.4 | 300.0 |
| **525 SD** | **720** | **480** | 393 216 | - | - | - | 42.1 | 84.3 | 170.0 | 300.0 |
| **4CIF** | **704** | **576** | 405 504 | - | - | - | 40.9 | 81.8 | 164.8 | 300.0 |
| **625 SD** | **720** | **576** | 442 368 | - | - | - | 37.5 | 75.0 | 151.1 | 300.0 |
| **480p (16:9)** | **864** | **480** | 458 752 | - | - | - | 36.1 | 72.3 | 145.7 | 291.4 |
| **SVGA** | **800** | **600** | 532 480 | - | - | - | 31.1 | 62.3 | 125.5 | 251.0 |
| **QHD** | **960** | **540** | 552 960 | - | - | - | 30.0 | 60.0 | 120.8 | 241.7 |
| **XGA** | **1 024** | **768** | 786 432 | - | - | - | - | 42.1 | 85.0 | 170.0 |
| **720p HD** | **1 280** | **720** | 983 040 | - | - | - | - | 33.7 | 68.0 | 136.0 |
| **4VGA** | **1 280** | **960** | 1 228 800 | - | - | - | - | - | 54.4 | 108.8 |
| **SXGA** | **1 280** | **1 024** | 1 310 720 | - | - | - | - | - | 51.0 | 102.0 |
| **525 16SIF** | **1 408** | **960** | 1 351 680 | - | - | - | - | - | 49.4 | 98.9 |
| **16CIF** | **1 408** | **1 152** | 1 622 016 | - | - | - | - | - | 41.2 | 82.4 |
| **4SVGA** | **1 600** | **1 200** | 1 945 600 | - | - | - | - | - | 34.3 | 68.7 |
| **1080 HD** | **1 920** | **1 080** | 2 088 960 | - | - | - | - | - | 32.0 | 64.0 |
| **2Kx1K** | **2 048** | **1 024** | 2 097 152 | - | - | - | - | - | 31.8 | 63.7 |
| **2Kx1080** | **2 048** | **1 080** | 2 228 224 | - | - | - | - | - | 30.0 | 60.0 |
| **4XGA** | **2 048** | **1 536** | 3 145 728 | - | - | - | - | - | - | - |
| **16VGA** | **2 560** | **1 920** | 4 915 200 | - | - | - | - | - | - | - |
| **3616x1536 (2.35:1)** | **3 616** | **1 536** | 5 603 328 | - | - | - | - | - | - | - |
| **3672x1536 (2.39:1)** | **3 680** | **1 536** | 5 701 632 | - | - | - | - | - | - | - |
| **3840x2160 (4\*HD)** | **3 840** | **2 160** | 8 355 840 | - | - | - | - | - | - | - |
| **4Kx2K** | **4 096** | **2 048** | 8 388 608 | - | - | - | - | - | - | - |
| **4096x2160** | **4 096** | **2 160** | 8 912 896 | - | - | - | - | - | - | - |
| **4096x2304 (16:9)** | **4 096** | **2 304** | 9 437 184 | - | - | - | - | - | - | - |
| **7680x4320** | **7 680** | **4 320** | 33 423 360 | - | - | - | - | - | - | - |
| **8192x4096** | **8 192** | **4 096** | 33 554 432 | - | - | - | - | - | - | - |
| **8192x4320** | **8 192** | **4 320** | 35 651 584 | - | - | - | - | - | - | - |
| **11520x6480** | **11 520** | **6 480** | 74 649 600 | - | - | - | - | - | - | - |
| **12288x6144** | **12 288** | **6 144** | 75 497 472 | - | - | - | - | - | - | - |
| **12288x6480** | **12 288** | **6 480** | 79 626 240 | - | - | - | - | - | - | - |
| **15360x8640** | **15 360** | **8 640** | 132 710 400 | - | - | - | - | - | - | - |
| **16384x8192** | **16 384** | **8 192** | 134 217 728 | - | - | - | - | - | - | - |
| **16384x8640** | **16 384** | **8 640** | 141 557 760 | - | - | - | - | - | - | - |

**Table****A.12 – Maximum picture rates (pictures per second) at levels 5 to 6.3 for some example picture sizes  
when MinCbSizeY is equal to 64**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Level:** |  |  |  | **5** | **5.1** | **5.2** | **6** | **6.1** | **6.2** | **6.3** |
| **Max luma picture size (samples):** |  |  |  | 8 912 896 | 8 912 896 | 8 912 896 | 35 651 584 | 35 651 584 | 35 651 584 | 80 216 064 |
| **Max luma sample rate (samples/sec)** |  |  |  | 267 386 880 | 534 773 760 | 1 069 547 520 | 1 069 547 520 | 2 139 095 040 | 4 278 190 080 | 4 812 963 840 |
| **Format nickname** | **Luma width** | **Luma height** | **Luma picture size** |  |  |  |  |  |  |  |
| **SQCIF** | **128** | **96** | 16 384 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **QCIF** | **176** | **144** | 36 864 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **QVGA** | **320** | **240** | 81 920 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **525 SIF** | **352** | **240** | 98 304 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **CIF** | **352** | **288** | 122 880 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **525 HHR** | **352** | **480** | 196 608 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **625 HHR** | **352** | **576** | 221 184 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **Q720p** | **640** | **360** | 245 760 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **VGA** | **640** | **480** | 327 680 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **525 4SIF** | **704** | **480** | 360 448 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **525 SD** | **720** | **480** | 393 216 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **4CIF** | **704** | **576** | 405 504 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **625 SD** | **720** | **576** | 442 368 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **480p (16:9)** | **864** | **480** | 458 752 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **SVGA** | **800** | **600** | 532 480 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **QHD** | **960** | **540** | 552 960 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **XGA** | **1 024** | **768** | 786 432 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **720p HD** | **1 280** | **720** | 983 040 | 272.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **4VGA** | **1 280** | **960** | 1 228 800 | 217.6 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **SXGA** | **1 280** | **1 024** | 1 310 720 | 204.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **525 16SIF** | **1 408** | **960** | 1 351 680 | 197.8 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **16CIF** | **1 408** | **1 152** | 1 622 016 | 164.8 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **4SVGA** | **1 600** | **1 200** | 1 945 600 | 137.4 | 274.8 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **1080 HD** | **1 920** | **1 080** | 2 088 960 | 128.0 | 256.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **2Kx1K** | **2 048** | **1 024** | 2 097 152 | 127.5 | 255.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **2Kx1080** | **2 048** | **1 080** | 2 228 224 | 120.0 | 240.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **4XGA** | **2 048** | **1 536** | 3 145 728 | 85.0 | 170.0 | 300.0 | 300.0 | 300.0 | 300.0 | 300.0 |
| **16VGA** | **2 560** | **1 920** | 4 915 200 | 54.4 | 108.8 | 217.6 | 217.6 | 300.0 | 300.0 | 300.0 |
| **3616x1536 (2.35:1)** | **3 616** | **1 536** | 5 603 328 | 47.7 | 95.4 | 190.8 | 190.8 | 300.0 | 300.0 | 300.0 |
| **3672x1536 (2.39:1)** | **3 680** | **1 536** | 5 701 632 | 46.8 | 93.7 | 187.5 | 187.5 | 300.0 | 300.0 | 300.0 |
| **3840x2160 (4\*HD)** | **3 840** | **2 160** | 8 355 840 | 32.0 | 64.0 | 128.0 | 128.0 | 256.0 | 300.0 | 300.0 |
| **4Kx2K** | **4 096** | **2 048** | 8 388 608 | 31.8 | 63.7 | 127.5 | 127.5 | 255.0 | 300.0 | 300.0 |
| **4096x2160** | **4 096** | **2 160** | 8 912 896 | 30.0 | 60.0 | 120.0 | 120.0 | 240.0 | 300.0 | 300.0 |
| **4096x2304 (16:9)** | **4 096** | **2 304** | 9 437 184 | - | - | - | 113.3 | 226.6 | 300.0 | 300.0 |
| **4096x3072** | **4 096** | **3 072** | 12 582 912 | - | - | - | 85.0 | 170.0 | 300.0 | 300.0 |
| **7680x4320** | **7 680** | **4 320** | 33 423 360 | - | - | - | 32.0 | 64.0 | 128.0 | 144.0 |
| **8192x4096** | **8 192** | **4 096** | 33 554 432 | - | - | - | 31.8 | 63.7 | 127.5 | 143.4 |
| **8192x4320** | **8 192** | **4 320** | 35 651 584 | - | - | - | 30.0 | 60.0 | 120.0 | 135.0 |
| **11520x6480** | **11 520** | **6 480** | 74 649 600 | - | - | - | - | - | - | 64.0 |
| **12288x6144** | **12 288** | **6 144** | 75 497 472 | - | - | - | - | - | - | 63.7 |
| **12288x6480** | **12 288** | **6 480** | 79 626 240 | - | - | - | - | - | - | 60.0 |
| **15360x8640** | **15 360** | **8 640** | 132 710 400 | - | - | - | - | - | - | - |
| **16384x8192** | **16 384** | **8 192** | 134 217 728 | - | - | - | - | - | - | - |
| **16384x8640** | **16 384** | **8 640** | 141 557 760 | - | - | - | - | - | - | - |

*In subclause A.4.3, add Table A.13 as follows:*

**Table A.13 – Maximum picture rates (pictures per second) at levels 7 to 7.2 for some example picture sizes when MinCbSizeY is equal to 64**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Level:** |  |  |  | **7** | **7.1** | **7.2** |
| **Max luma picture size (samples):** |  |  |  | 142 606 336 | 142 606 336 | 142 606 336 |
| **Max luma sample rate (samples/sec)** |  |  |  | 4 812 963 840 | 8 556 380 160 | 17 112 760 320 |
| **Format nickname** | **Luma width** | **Luma height** | **Luma picture size** |  |  |  |
| **SQCIF** | **128** | **96** | 16 384 | 300.0 | 300.0 | 300.0 |
| **QCIF** | **176** | **144** | 36 864 | 300.0 | 300.0 | 300.0 |
| **QVGA** | **320** | **240** | 81 920 | 300.0 | 300.0 | 300.0 |
| **525 SIF** | **352** | **240** | 98 304 | 300.0 | 300.0 | 300.0 |
| **CIF** | **352** | **288** | 122 880 | 300.0 | 300.0 | 300.0 |
| **525 HHR** | **352** | **480** | 196 608 | 300.0 | 300.0 | 300.0 |
| **625 HHR** | **352** | **576** | 221 184 | 300.0 | 300.0 | 300.0 |
| **Q720p** | **640** | **360** | 245 760 | 300.0 | 300.0 | 300.0 |
| **VGA** | **640** | **480** | 327 680 | 300.0 | 300.0 | 300.0 |
| **525 4SIF** | **704** | **480** | 360 448 | 300.0 | 300.0 | 300.0 |
| **525 SD** | **720** | **480** | 393 216 | 300.0 | 300.0 | 300.0 |
| **4CIF** | **704** | **576** | 405 504 | 300.0 | 300.0 | 300.0 |
| **625 SD** | **720** | **576** | 442 368 | 300.0 | 300.0 | 300.0 |
| **480p (16:9)** | **864** | **480** | 458 752 | 300.0 | 300.0 | 300.0 |
| **SVGA** | **800** | **600** | 532 480 | 300.0 | 300.0 | 300.0 |
| **QHD** | **960** | **540** | 552 960 | 300.0 | 300.0 | 300.0 |
| **XGA** | **1 024** | **768** | 786 432 | 300.0 | 300.0 | 300.0 |
| **720p HD** | **1 280** | **720** | 983 040 | 300.0 | 300.0 | 300.0 |
| **4VGA** | **1 280** | **960** | 1 228 800 | 300.0 | 300.0 | 300.0 |
| **SXGA** | **1 280** | **1 024** | 1 310 720 | 300.0 | 300.0 | 300.0 |
| **525 16SIF** | **1 408** | **960** | 1 351 680 | 300.0 | 300.0 | 300.0 |
| **16CIF** | **1 408** | **1 152** | 1 622 016 | 300.0 | 300.0 | 300.0 |
| **4SVGA** | **1 600** | **1 200** | 1 945 600 | 300.0 | 300.0 | 300.0 |
| **1080 HD** | **1 920** | **1 080** | 2 088 960 | 300.0 | 300.0 | 300.0 |
| **2Kx1K** | **2 048** | **1 024** | 2 097 152 | 300.0 | 300.0 | 300.0 |
| **2Kx1080** | **2 048** | **1 080** | 2 228 224 | 300.0 | 300.0 | 300.0 |
| **4XGA** | **2 048** | **1 536** | 3 145 728 | 300.0 | 300.0 | 300.0 |
| **16VGA** | **2 560** | **1 920** | 4 915 200 | 300.0 | 300.0 | 300.0 |
| **3616x1536 (2.35:1)** | **3 616** | **1 536** | 5 603 328 | 300.0 | 300.0 | 300.0 |
| **3672x1536 (2.39:1)** | **3 680** | **1 536** | 5 701 632 | 300.0 | 300.0 | 300.0 |
| **3840x2160 (4\*HD)** | **3 840** | **2 160** | 8 355 840 | 300.0 | 300.0 | 300.0 |
| **4Kx2K** | **4 096** | **2 048** | 8 388 608 | 300.0 | 300.0 | 300.0 |
| **4096x2160** | **4 096** | **2 160** | 8 912 896 | 300.0 | 300.0 | 300.0 |
| **4096x2304 (16:9)** | **4 096** | **2 304** | 9 437 184 | 300.0 | 300.0 | 300.0 |
| **4096x3072** | **4 096** | **3 072** | 12 582 912 | 300.0 | 300.0 | 300.0 |
| **7680x4320** | **7 680** | **4 320** | 33 423 360 | 144.0 | 256.0 | 300.0 |
| **8192x4096** | **8 192** | **4 096** | 33 554 432 | 143.4 | 255.0 | 300.0 |
| **8192x4320** | **8 192** | **4 320** | 35 651 584 | 135.0 | 240.0 | 300.0 |
| **11520x6480** | **11 520** | **6 480** | 74 649 600 | 64.0 | 114.6 | 229.2 |
| **12288x6144** | **12 288** | **6 144** | 75 497 472 | 63.7 | 113.3 | 226.6 |
| **12288x6480** | **12 288** | **6 480** | 79 626 240 | 60.0 | 107.4 | 214.9 |
| **15360x8640** | **15 360** | **8 640** | 132 710 400 | 36.2 | 64.4 | 128.8 |
| **16384x8192** | **16 384** | **8 192** | 134 217 728 | 35.8 | 63.7 | 127.5 |
| **16384x8640** | **16 384** | **8 640** | 141 557 760 | 34.0 | 60.4 | 120.8 |

*In subclause C.2.3, make the following changes:*

The variables InitCpbRemovalDelay[ SchedSelIdx ] and InitCpbRemovalDelayOffset[ SchedSelIdx ] are updated, and the variables CpbDelayOffset and DpbDelayOffset are derived, as follows:

...

if( !concatenationFlag ) {  
 baseTime = AuNominalRemovalTime[ firstPicInPrevBuffPeriod ]  
 tmpCpbRemovalDelay = AuCpbRemovalDelayVal  
 tmpCpbDelayOffset = CpbDelayOffset  
} else {  
 baseTime1 = AuNominalRemovalTime[ prevNonDiscardablePic ]  
 ~~tmpCpbRemovalDelay =  
 Max( ( auCpbRemovalDelayDeltaMinus1 + 1 ),  
 Ceil( ( InitCpbRemovalDelay[ SchedSelIdx ] ÷ 90 000 +  
 AuFinalArrivalTime[ n − 1 ] − AuNominalRemovalTime[ n − 1 ] ) ÷ ClockTick ) )~~ 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  
 }  
 tmpCpbDelayOffset = 0  
}  
AuNominalRemovalTime[ n ] = baseTime + ClockTick \* ( tmpCpbRemovalDelay − tmpCpbDelayOffset )

...

*In subclause C.4, make the following changes:*

...

6. For each current picture, after invocation of the process for removal of pictures from the DPB as specified in C.3.2, the number of decoded pictures in the DPB, including all pictures n that are marked as “used for reference”, or that have PicOutputFlag equal to 1 and ~~AuCpbRemovalTime[ n ]~~ DpbOutputTime[ n ] less than AuCpbRemovalTime[ currPic ], where currPic is the current picture, shall be less than or equal to sps\_max\_dec\_pic\_buffering\_minus1[ HighestTid ].

...

*In subclause D.2.2, make the following changes:*

|  |  |
| --- | --- |
| buffering\_period( payloadSize ) { | **Descriptor** |
| **...** |  |
| if( more\_data\_in\_payload( ) ) |  |
| if( payload\_extension\_present( ) ) |  |
| **use\_alt\_cpb\_params\_flag** | u(1) |
| } |  |

*In subclause D.2.47, make the following changes:*

|  |  |
| --- | --- |
| annotated\_regions( payloadSize ) { | **Descriptor** |
| **...** |  |
| for( i = 0; i < ar\_num\_object\_updates; i++ ) { |  |
| ~~for( i = 0; i  <=  ar\_num\_object\_updates; i++ ) {~~ |  |
| **...** |  |
| } |  |

*In subclause D.3.1, make the following changes:*

...

|  |  |
| --- | --- |
| SEI manifest | The ~~CLVS~~ CVS containing the SEI message |
| SEI prefix indication | The ~~CLVS~~ CVS containing the SEI message |

...

The following applies on the applicable operation points or layers of SEI messages:

— For a non-scalable-nested SEI message, when payloadType is equal to 0 (buffering period) or 130 (decoding unit information), the non-scalable-nested SEI message applies to the operation point that has OpTid equal to the greatest value of nuh\_temporal\_id\_plus1 among all VCL NAL units in the bitstream, ~~and that~~ has OpLayerIdList containing all values of nuh\_layer\_id in all VCL units in the bitstream, and has only the base layer as the output layer.

— An SEI message that is directly contained in a scalable nesting SEI message within an SEI NAL unit with nuh\_layer\_id equal to 0 and has payloadType is equal to 0 (buffering period), 1 (picture timing), or 130 (decoding unit information) applies as specified in Annex C to the layer set as indicated by the scalable nesting SEI message.

— For a non-scalable-nested SEI message, when payloadType is equal to 1 (picture timing), the frame field information carried in the syntax elements pic\_struct, source\_scan\_type, and duplicate\_flag, when present, in the non-scalable-nested picture timing SEI message applies to the base layer only, while the picture timing information carried in other syntax elements, when present, in the non-scalable-nested picture timing SEI message applies to the operation point that has OpTid equal to the greatest value of nuh\_temporal\_id\_plus1 among all VCL NAL units in the bitstream, ~~and that~~ has OpLayerIdList containing all values of nuh\_layer\_id in all VCL units in the bitstream, and has only the base layer as the output layer.

...

*In subclause D.3.2, change the semantics of the buffering period SEI message as follows:*

...

**use\_alt\_cpb\_params\_flag** may be used to derive the value of UseAltCpbParamsFlag. When irap\_cpb\_params\_present\_flag is equal to 0, use\_alt\_cpb\_params\_flag shall not be equal to 1. When use\_alt\_cpb\_params\_flag is not present, it is inferred to be equal to 0.

NOTE 4 – The syntax element use\_alt\_cpb\_params\_flag may be present in the payload extension of the buffering period SEI message. Decoders conforming to profiles specified in Annex A may ~~ingore~~ ignore this syntax element.

It is a requirement of bitstream conformance that when use\_alt\_cpb\_params\_flag is present in the buffering perid SEI message, the return value of the more\_data\_in\_payload( ) function in the sei\_payload( ) syntax structure containing the buffering period SEI message shall be equal to 1.

*In subclause D.3.20, make the following changes:*

...

where component[ cIdx ][ i ] is ~~an~~ a 2-dimension array ~~in raster scan~~ of the decoded sample values ~~in two's complement representation~~ of a component of a decoded picture.

...

*In subclause D.3.13, make the following changes:*

This SEI message provides the decoder with a parameterized model for film grain synthesis.

NOTE 1 – For example, an encoder could use the film grain characteristics SEI message to characterize film grain that was present in the original source video material and was removed by pre-processing filtering techniques. Synthesis of simulated film grain on the ~~decoded images~~ input images, which may be the decoded pictures or converted from the decoded pictures, for the display process is optional and does not need to exactly follow the specified semantics of the film grain characteristics SEI message. When synthesis of simulated film grain on the ~~decoded images~~ input images for the display process is performed, there is no requirement that the method by which the synthesis is performed be the same as the parameterized model for the film grain as provided in the film grain characteristics SEI message.

NOTE 2 – The display process is not specified in this Specification.

NOTE 3 – SMPTE RDD 5 specifies a film grain simulator based on the information provided in the film grain characteristics SEI message.

The film grain models specified in the film grain characteristics SEI message are expressed for application to decoded pictures that have 4:4:4 colour format with luma and chroma bit depths corresponding to the luma and chroma bit depths of the film grain model and use the same colour representation domain as the identified film grain model. When the colour format of the decoded video is not 4:4:4 or the decoded video uses a different luma or chroma bit depth from that of the film grain model or uses a different colour representation domain from that of the identified film grain model, an unspecified conversion process is expected to be applied to convert the decoded pictures to the form that is expressed for application of the film grain model.

NOTE 4 – Because the use of a specific method is not required for performing the film grain generation function used by the display process, a decoder could, if desired, down-convert the model information for chroma in order to simulate film grain for other chroma formats (4:2:0 or 4:2:2) rather than up-converting the decoded video (using a method not specified in this Specification) before performing film grain generation.

**film\_grain\_characteristics\_cancel\_flag** equal to 1 indicates that the SEI message cancels the persistence of any previous film grain characteristics SEI message in output order that applies to the current layer. film\_grain\_characteristics\_cancel\_flag equal to 0 indicates that film grain modelling information follows.

**film\_grain\_model\_id** identifies the film grain simulation model as specified in Table D.4. The value of film\_grain\_model\_id shall be in the range of 0 to 1, inclusive. The values of 2 and 3 for film\_grain\_model\_id are reserved for future use by ITU‑T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders shall ignore film grain characteristic SEI messages with film\_grain\_model\_id equal to 2 or 3.

**Table D.4 – film\_grain\_model\_id values**

|  |  |
| --- | --- |
| **Value** | **Description** |
| 0 | Frequency filtering |
| 1 | Auto-regression |

**separate\_colour\_description\_present\_flag** equal to 1 indicates that a distinct combination of luma bit depth, chroma bit depth, video full range flag, colour primaries, transfer characteristics, and matrix coefficients for the film grain characteristics specified in the SEI message is present in the film grain characteristics SEI message syntax. separate\_colour\_description\_present\_flag equal to 0 indicates that the combination of luma bit depth, chroma bit depth, video full range flag, colour primaries, transfer characteristics, and matrix coefficients for the film grain characteristics specified in the SEI message are the same as indicated in VUI parameters for the CVS.

NOTE 5 – When separate\_colour\_description\_present\_flag is equal to 1, any of the luma bit depth, chroma bit depth, video full range flag, colour primaries, transfer characteristics, and matrix coefficients specified for the film grain characteristics specified in the SEI message could differ from that for the pictures in the CVS.

When VUI parameters are not present for the CVS or the value of colour\_description\_present\_flag is equal to 0, and equivalent information to that conveyed when colour\_description\_present\_flag is equal to 1 is not conveyed by external means, separate\_colour\_description\_present\_flag shall be equal to 1.

The input image Î, which may be the decoded picture or converted from the decoded picture, used in the equations in this clause is in the same colour representation domain as the simulated film grain signal. Therefore, when any of these parameters does differ from that for the pictures in the CVS, the input image Î used in the equations in this clause would be in a different colour representation domain than that for the pictures in the CVS. For example, when the value of film\_grain\_bit\_depth\_luma\_minus8 + 8 is greater than the bit depth of the luma component of the pictures in the CVS, the bit depth of the input image Î used in the equations in this clause is also greater than the bit depth of the luma component of the pictures in the CVS. In such a case, the input image Î would be generated by converting the actual decoded picture to be in the same colour representation domain as the simulated film grain signal. The process for converting an actual decoded pictures to the 4:4:4 colour format with same colour representation domain as the simulated film grain signal is not specified in this Specification.

**film\_grain\_bit\_depth\_luma\_minus8** plus 8 specifies the bit depth used for the luma component of the film grain characteristics specified in the SEI message. When film\_grain\_bit\_depth\_luma\_minus8 is not present in the film grain characteristics SEI message, the value of film\_grain\_bit\_depth\_luma\_minus8 is inferred to be equal to bit\_depth\_luma\_minus8.

The value of filmGrainBitDepth[ 0 ] is derived as follows:

filmGrainBitDepth[ 0 ] = film\_grain\_bit\_depth\_luma\_minus8 + 8 (D-3)

**film\_grain\_bit\_depth\_chroma\_minus8** plus 8 specifies the bit depth used for the Cb and Cr components of the film grain characteristics specified in the SEI message. When film\_grain\_bit\_depth\_chroma\_minus8 is not present in the film grain characteristics SEI message, the value of film\_grain\_bit\_depth\_chroma\_minus8 is inferred to be equal to bit\_depth\_chroma\_minus8.

The value of filmGrainBitDepth[ c ] for c = 1 and 2 is derived as follows:

filmGrainBitDepth[ c ] = film\_grain\_bit\_depth\_chroma\_minus8 + 8, with c = 1, 2 (D-4)

**film\_grain\_full\_range\_flag** has the same semantics as specified in clause E.3.1 for the video\_full\_range\_flag syntax element, except as follows:

– film\_grain\_full\_range\_flag specifies the video full range flag of the film grain characteristics specified in the SEI message, rather than the video full range flag used for the CVS.

– When film\_grain\_full\_range\_flag is not present in the film grain characteristics SEI message, the value of film\_grain\_full\_range\_flag is inferred to be equal to video\_full\_range\_flag.

**film\_grain\_colour\_primaries** has the same semantics as specified in clause E.3.1 for the colour\_primaries syntax element, except as follows:

– film\_grain\_colour\_primaries specifies the colour primaries of the film grain characteristics specified in the SEI message, rather than the colour primaries used for the CVS.

– When film\_grain\_colour\_primaries is not present in the film grain characteristics SEI message, the value of film\_grain\_colour\_primaries is inferred to be equal to colour\_primaries.

**film\_grain\_transfer\_characteristics** has the same semantics as specified in clause E.3.1 for the transfer\_characteristics syntax element, except as follows:

– film\_grain\_transfer\_characteristics specifies the transfer characteristics of the film grain characteristics specified in the SEI message, rather than the transfer characteristics used for the CVS.

– When film\_grain\_transfer\_characteristics is not present in the film grain characteristics SEI message, the value of film\_grain\_transfer\_characteristics is inferred to be equal to transfer\_characteristics.

**film\_grain\_matrix\_coeffs** has the same semantics as specified in clause E.3.1 for the matrix\_coeffs syntax element, except as follows:

– film\_grain\_matrix\_coeffs specifies the matrix coefficients of the film grain characteristics specified in the SEI message, rather than the matrix coefficients used for the CVS.

– When film\_grain\_matrix\_coeffs is not present in the film grain characteristics SEI message, the value of film\_grain\_matrix\_coeffs is inferred to be equal to matrix\_coeffs.

– The values allowed for film\_grain\_matrix\_coeffs are not constrained by the chroma\_format\_idc of the decoded pictures that is indicated by the value of chroma\_format\_idc for the semantics of the VUI parameters.

**blending\_mode\_id** identifies the blending mode used to blend the simulated film grain with the ~~decoded images~~ input images as specified in Table D.5. blending\_mode\_id shall be in the range of 0 to 1, inclusive. The values of 2 and 3 for blending\_mode\_id are reserved for future use by ITU‑T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders shall ignore film grain characteristic SEI messages with blending\_mode\_id equal to 2 or 3.

**Table D.5 – blending\_mode\_id values**

|  |  |
| --- | --- |
| **Value** | **Description** |
| 0 | Additive |
| 1 | Multiplicative |

Depending on the value of blending\_mode\_id, the blending mode is specified as follows:

– If blending\_mode\_id is equal to 0, the blending mode is additive as specified by:

Igrain[ c ][ x ][ y ] = Clip3( 0, ( 1 << filmGrainBitDepth[ c ] ) − 1, ~~I~~~~decoded~~Î[ c ][ x ][ y ] + G[ c ][ x ][ y ] ) (D-5)

– Otherwise (blending\_mode\_id is equal to 1), the blending mode is multiplicative as specified by:

Igrain[ c ][ x ][ y ] = Clip3( 0, ( 1 << filmGrainBitDepth[ c ] ) − 1, ~~I~~~~decoded~~Î[ c ][ x ][ y ] + (D-6)  
 Round( ( ~~I~~~~decoded~~Î[ c ][ x ][ y ] \* G[ c ][ x ][ y ] ) ÷ ( ( 1  <<  bitDepth[ c ] ) − 1 ) ) )

where ~~I~~~~decoded~~Î[ c ][ x ][ y ] represents the sample value at coordinates x, y of the colour component c of the ~~decoded image I~~~~decoded~~input image Î, G[ c ][ x ][ y ] is the simulated film grain value at the same position and colour component, and filmGrainBitDepth[ c ] is the number of bits used for each sample in a fixed-length unsigned binary representation of the arrays Igrain[ c ][ x ][ y ], Î[ c ][ x ][ y ], and G[ c ][ x ][ y ], where c = 0..2, x = 0..pic\_width\_in\_luma\_samples − 1, and y = 0..pic\_height\_in\_luma\_samples – 1

**log2\_scale\_factor** specifies a scale factor used in the film grain characterization equations.

**comp\_model\_present\_flag**[ c ]equal to 0 indicates that film grain is not modelled on the c-th colour component, where c equal to 0 refers to the luma component, c equal to 1 refers to the Cb component, and c equal to 2 refers to the Cr component. comp\_model\_present\_flag[ c ] equal to 1 indicates that syntax elements specifying modelling of film grain on colour component c are present in the SEI message.

When separate\_colour\_description\_present\_flag is equal to 0 and chroma\_format\_idc is equal to 0, the value of comp\_model\_present\_flag[ 1 ] and comp\_model\_present\_flag[ 2 ] shall be equal to 0.

**num\_intensity\_intervals\_minus1**[ c ] plus 1 specifies the number of intensity intervals for which a specific set of model values has been estimated.

NOTE 6 – The intensity intervals could overlap in order to simulate multi-generational film grain.

**num\_model\_values\_minus1**[ c ] plus 1 specifies the number of model values present for each intensity interval in which the film grain has been modelled. The value of num\_model\_values\_minus1[ c ] shall be in the range of 0 to 5, inclusive.

**intensity\_interval\_lower\_bound**[ c ][ i ] specifies the lower bound of the i-th intensity interval for which the set of model values applies.

**intensity\_interval\_upper\_bound**[ c ][ i ] specifies the upper bound of the i-th intensity interval for which the set of model values applies.

The variable intensityIntervalIdx[ c ][ x ][ y ][ j ] represents the j-th index to the list of intensity intervals selected for the sample value Î[ c ][ x ][ y ] for c = 0..2, x = 0..pic\_width\_in\_luma\_samples − 1, y = 0..pic\_height\_in\_luma\_samples − 1, and j = 0..numApplicableIntensityIntervals[ c ][ x ][ y ] − 1, where numApplicableIntensityIntervals[ c ][ x ][ y ] is derived below.

Depending on the value of film\_grain\_model\_id, the selection of one or more intensity intervals for the sample value Î[ c ][ x ][ y ] is specified as follows:

– The variable numApplicableIntensityIntervals[ c ][ x ][ y ] is initially set equal to 0.

– If film\_grain\_model\_id is equal to 0, the following applies:

* The top-left sample location ( xB, yB ) of the current 8x8 block b that contains the sample value Î[ c ][ x ][ y ] is derived as ( xB, yB ) = ( x / 8, y / 8 ).
* The average value bavg of the current 8x8 block is derived as follows:

sum8x8 = 0  
for( i = 0; i  < 8; i++ )  
 for( j = 0; j < 8; j++ )  
      sum8x8 += Î[ xB \* 8 + I, yB \* 8 + j, c ] (D-7)  
bavg = Clip3( 0, 255, ( sum8x8 + ( 1  <<  ( filmGrainBitDepth[ c ] − 3 ) ) )  >>  ( filmGrainBitDepth[ c ] − 2 ) )

* The value of intensityIntervalIdx[ c ][ x ][ y ][ j ] is derived as follows:

for( i = 0, j = 0; i  <=  num\_intensity\_intervals\_minus1[ c ]; i++ )  
 if( bavg  >=  intensity\_interval\_lower\_bound[ c ][ i ]    
 &&  bavg  <=  intensity\_interval\_upper\_bound[ c ][ i ] ) {  
 intensityIntervalIdx[ c ][ x ][ y ][ j ] = i (D-8)  
 j++  
 }  
numApplicableIntensityIntervals[ c ][ x ][ y ] = j

– Otherwise (film\_grain\_model\_id is equal to 1), the value of intensityIntervalIdx[ c ][ x ][ y ][ j ] is derived as follows:

I8[ c ][ x ][ y ] = ( filmGrainBitDepth[ c ]  = =  8 ) ? ( Î[ c ][ x ][ y ] :  
 Clip3( 0, 255, ( Î[ c ][ x ][ y ] + ( 1  <<  ( filmGrainBitDepth[ c ] − 9 ) ) )  >>  ( filmGrainBitDepth[ c ] − 8 ) )  
for( i = 0, j = 0; i  <=  num\_intensity\_intervals\_minus1[ c ]; i++ )  
 if( I8[ c ][ x ][ y ]  >=  intensity\_interval\_lower\_bound[ c ][ i ]  &&    
 I8[ c ][ x ][ y ]  <=  intensity\_interval\_upper\_bound[ c ][ i ] ) { (D-9)  
 intensityIntervalIdx[ c ][ x ][ y ][ j ] = i  
 j++  
 }  
numApplicableIntensityIntervals[ c ][ x ][ y ] = j

Samples that do not fall into any of the defined intervals (i.e., those samples for which the value of numApplicableIntensityIntervals[ c ][ x ][ y ] is equal to 0) are not modified by the grain generation function. Samples that fall into more than one interval (i.e., those samples for which the value of numApplicableIntensityIntervals[ c ][ x ][ y ] is greater than 1) will originate multi-generation grain. Multi-generation grain results from adding the grain computed independently for each of the applicable intensity intervals.

In the equations in the remainder of this clause, the variable sj in each instance of the list comp\_model\_value[ c ][ sj ] is the value of intensityIntervalIdx[ c ][ x ][ y ][ j ] derived for the sample value Î[ c ][ x ][ y ].

**comp\_model\_value**[ c ][ i ][ j ] specifies the j-th model value present for the colour component c and the i-th intensity interval. The set of model values has different meaning depending on the value of film\_grain\_model\_id.

The value of comp\_model\_value[ c ][ i ][ j ] is constrained as follows, and could be additionally constrained as specified elsewhere in this clause:

– If film\_grain\_model\_id is equal to 0, comp\_model\_value[ c ][ i ][ j ] shall be in the range of 0 to 2filmGrainBitDepth[ c ] − 1, inclusive.

– Otherwise (film\_grain\_model\_id is equal to 1), comp\_model\_value[ c ][ i ][ j ] shall be in the range of −2( filmGrainBitDepth[ c ] − 1 ) to 2( filmGrainBitDepth[ c ] − 1 ) − 1, inclusive.

Depending on the value of film\_grain\_model\_id, the synthesis of the film grain is modelled as follows:

– If film\_grain\_model\_id is equal to 0, a frequency filtering model enables simulating the original film grain for c = 0..2, x = 0..pic\_width\_in\_luma\_samples − 1 and y = 0..pic\_height\_in\_luma\_samples − 1 as specified by:

G[ c ][ x ][ y ] = ( comp\_model\_value[ c ][ sj ][ 0 ] \* Q[ c ][ x ][ y ] + comp\_model\_value[ c ][ sj ][ 5 ] \*  
 G[ c − 1 ][ x ][ y ] ) >> log2\_scale\_factor (D-10)

where Q[ c ] is a two-dimensional random process generated by filtering 16x16 blocks gaussRv with random-value elements gaussRvij generated with a normalized Gaussian distribution (independent and identically distributed Gaussian random variable samples with zero mean and unity variance) and where the value of an element G[ c − 1 ][ x ][ y ] used in the right-hand side of the equation is inferred to be equal to 0 when c − 1 is less than 0.

NOTE 7 – A normalized Gaussian random variable can be generated from two independent, uniformly distributed random values over the interval from 0 to 1 (and not equal to 0), denoted as uRv0 and uRv1, using the Box-Muller transformation specified by:

gaussRvi,j = Sqrt( −2 \* Ln( uRv0 ) ) \* Cos( 2 \* π \* uRv1 ) (D-11)

where π is Archimedes' constant 3.141 592 653 589 793....

The band-pass filtering of blocks gaussRv can be performed in the discrete cosine transform (DCT) domain as follows:

for( y = 0; y < 16; y++ )  
 for( x = 0; x < 16; x++ )  
 if( ( x < comp\_model\_value[ c ][ sj ][ 3 ]  &&  y < comp\_model\_value[ c ][ sj ][ 4 ] )  | | (D-12)  
 x > comp\_model\_value[ c ][ sj ][ 1 ]  | |  y > comp\_model\_value[ c ][ sj ][ 2 ] )  
 gaussRv[ x ][ y ] = 0  
filteredRv = IDCT16x16( gaussRv )

where IDCT16x16( z ) refers to a unitary inverse discrete cosine transformation (IDCT) operating on a 16x16 matrix argument z as specified by:

IDCT16x16( z ) = r \* z \* rT (D-13)

where the superscript T indicates a matrix transposition and r is the 16x16 matrix with elements rij specified by:

(D-14)

where π is Archimedes' constant 3.141 592 653 589 793....

Q[ c ] is formed by the frequency-filtered blocks filteredRv.

NOTE 8 – Coded model values are based on blocks of size 16x16, but a decoder implementation could use other block sizes. For example, decoders implementing the IDCT on 8x8 blocks could down-convert by a factor of two the set of coded model values comp\_model\_value[ c ][ sj ][ i ] for i equal to 1..4.

NOTE 9 – To reduce the degree of visible blocks can result from mosaicking the frequency-filtered blocks filteredRv, decoders could apply a low-pass filter to the boundaries between frequency-filtered blocks.

– Otherwise (film\_grain\_model\_id is equal to 1), an auto-regression model enables simulating the original film grain for c = 0..2, x = 0..pic\_width\_in\_luma\_samples – 1, and y = 0..pic\_height\_in\_luma\_samples − 1 as specified by:

G[ c ][ x ][ y ] = ( comp\_model\_value[ c ][ sj ][ 0 ] \* n[ c ][ x ][ y ] +  
comp\_model\_value[ c ][ sj ][ 1 ] \* ( G[ c ][ x − 1 ][ y ] + ( ( comp\_model\_value[ c ][ sj ][ 4 ] \* G[ c ][ x ][ y − 1 ] ) >>  
 log2\_scale\_factor ) ) +  
comp\_model\_value[ c ][ sj ][ 3 ] \* ( ( ( comp\_model\_value[ c ][ sj ][ 4 ] \* G[ c ][ x − 1 ][ y − 1 ] ) >>  
 log2\_scale\_factor ) + G[ c ][ x + 1 ][ y − 1 ] ) +  
comp\_model\_value[ c ][ sj ][ 5 ] \* ( G[ c ][ x − 2 ][ y ] +  
 ( ( comp\_model\_value[ c ][ sj ][ 4 ] \* comp\_model\_value[ c ][ sj ][ 4 ] \* G[ c ][ x ][ y − 2 ] ) >>  
 ( 2 \* log2\_scale\_factor ) ) ) +  
 comp\_model\_value[ c ][ sj ][ 2 ] \* G[ c − 1 ][ x ][ y ] ) >> log2\_scale\_factor (D-15)

where n[ c ][ x ][ y ] is a random value with normalized Gaussian distribution (independent and identically distributed Gaussian random variable samples with zero mean and unity variance for each value of x, y and c) and where the value of an element G[ c ][ x ][ y ] used in the right-hand side of the equation is inferred to be equal to 0 when any of the following conditions are true:

– x is less than 0,

– y is less than 0,

– c is less than 0.

...

*In subclause D.3.47, make the following changes:*

...

**ar\_label\_cancel\_flag** equal to 1 cancels the persistence scope of the ar\_label\_idx[ i ]-th label. ar\_label\_cancel\_flag equal to 0 indicates that the ar\_label\_idx[ i ]-th label is assigned a signalled value.

LabelAssigned[ ar\_label\_idx[ i ] ] equal to 1 indicates that the ar\_label\_idx[ i ]-th label is assigned. LabelAssigned[ ar\_label\_idx[ i ] ] equal to 0 indicates that the ar\_label\_idx[ i ]-th label is not assigned.

...

**ar\_object\_cancel\_flag** equal to 1 cancels the persistence scope of the ar\_object\_idx[ i ]-th object. ar\_object\_cancel\_flag equal to 0 indicates that parameters associated with the ar\_object\_idx[ i ]-th object tracked object are signalled.

ObjectTracked[ ar\_object\_idx[ i ] ] equal to 1 indicates that the object\_idx[ i ]-th object is tracked. ObjectTracked[ ar\_object\_idx[ i ] ] equal to 0 indicates that the object\_idx[ i ]-th object is not tracked.

**ar\_object\_label\_update\_flag** equal to 1 indicates that an object label is signalled. ar\_object\_label\_update\_flag equal to 0 indicates that an object label is not signalled.

**ar\_object\_label\_idx**[ ar\_object\_idx[ i ] ] indicates the index of the label corresponding to the ar\_object\_idx[ i ]-th object. When ar\_object\_label\_idx[ ar\_object\_idx[ i ] ] is not present, its value is inferred from a previous annotated regions SEI message in output order in the same CLVS, if any. The value of ar\_object\_label\_idx[ ar\_object\_idx[ i ] ] shall be in the range of 0 to 255, inclusive.

**ar\_bounding\_box\_update\_flag** equal to 1 indicates that object bounding box parameters are signalled. ar\_bounding\_box\_‌update\_flag equal to 0 indicates that object bounding box parameters are not signalled.

**ar\_bounding\_box\_cancel\_flag** equal to 1 cancels the persistence scope of the ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌width[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌height[ ar\_object\_idx[ i ] ]. ar\_partial\_object\_flag[ ar\_object\_idx[ i ] ], and ar\_object\_‌confidence[ ar\_object\_idx[ i ] ]. ar\_bounding\_box\_‌cancel\_flag equal to 0 indicates that ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌width[ ar\_object\_idx[ i ] ] ar\_bounding\_box\_‌height[ ar\_object\_idx[ i ] ] ar\_partial\_object\_flag[ ar\_object\_idx[ i ] ], and ar\_object\_‌confidence[ ar\_object\_idx[ i ] ] syntax elements are signalled.

ObjectBoundingBoxAvail[ ar\_object\_idx[ i ] ] equal to 1 indicates that the bounding box information of the object\_idx[ i ]-th object is signalled. ObjectBoundingBoxAvail[ ar\_object\_idx[ i ] ] equal to 0 indicates that the bounding box information of the object\_idx[ i ]-th object is not signalled.

**ar\_bounding\_box\_**‌**top**[ ar\_object\_idx[ i ] ], **ar\_bounding\_box\_**‌**left**[ ar\_object\_idx[ i ] ], **ar\_bounding\_box\_**‌**width**[ ar\_object\_idx[ i ] ], and **ar\_bounding\_box\_**‌**height**[ ar\_object\_idx[ i ] ] specify the coordinates of the top-left corner and the width and height, respectively, of the bounding box of the ar\_object\_idx[ i ]-th object in the cropped decoded picture, relative to the conformance cropping window specified by the active SPS.

Let croppedWidth and croppedHeight be the width and height, respectively, of the cropped decoded picture in units of luma samples, as specified by Equations D-28 and D-29.

The value of ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ] shall be in the range of 0 to croppedWidth / SubWidthC − 1, inclusive.

The value of ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ] shall be in the range of 0 to croppedHeight / SubHeightC − 1, inclusive.

The value of ar\_bounding\_box\_‌width[ ar\_object\_idx[ i ] ] shall be in the range of 0 to croppedWidth / SubWidthtC − ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ], inclusive.

The value of ar\_bounding\_box\_‌height[ ar\_object\_idx[ i ] ] shall be in the range of 0 to croppedHeight / SubHeightC − ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ], inclusive.

The identified object rectangle contains the luma samples with horizontal picture coordinates from SubWidthC \* ( conf\_win\_left\_offset + ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ] ) to SubWidthC \* ( conf\_win\_left\_offset + ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ] + ar\_bounding\_box\_‌width[ ar\_object\_idx[ i ] ] ) − 1, inclusive, and vertical picture coordinates from SubHeightC \* ( conf\_win\_top\_offset + ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ] ) to SubHeightC \* ( conf\_win\_top\_offset + ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ] + ar\_bounding\_box\_‌height[ ar\_object\_idx[ i ] ] ) − 1, inclusive.

When ChromaArrayType is not equal to 0, the corresponding specified samples of the two chroma arrays are the samples having picture coordinates ( x / SubWidthC, y / SubHeightC ), where ( x, y ) are the picture coordinates of the specified luma samples.

The values of ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌width[ ar\_object\_idx[ i ] ] and ar\_bounding\_box\_‌height[ ar\_object\_idx[ i ] ] persist in output order within the CLVS for each value of ar\_object\_idx[ i ]. When not present, the values of ar\_bounding\_box\_‌top[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌left[ ar\_object\_idx[ i ] ], ar\_bounding\_box\_‌width[ ar\_object\_idx[ i ] ] or ar\_bounding\_box\_‌height[ ar\_object\_idx[ i ] ] are inferred from a previous annotated regions SEI message in output order in the CLVS, if any.

...

*In subclause E.3.2, make the following changes:*

...

**elemental\_duration\_in\_tc\_minus1**[ i ] plus 1 (when present) specifies, when HighestTid is equal to i, the temporal distance, in clock ticks, between the elemental units that specify the HRD output times of consecutive pictures in output order as specified below. The value of elemental\_duration\_in\_tc\_minus1[ i ] shall be in the range of 0 to 2 047, inclusive.

When HighestTid is equal to i and fixed\_pic\_rate\_within\_cvs\_flag[ i ] is equal to 1 for a CVS containing picture n, and for ~~For~~ each picture n that is output and not the last picture in the bitstream (in output order) that is output, the value of the variable DpbOutputElementalInterval[ n ] is specified by:

DpbOutputElementalInterval[ n ] = DpbOutputInterval[ n ] ÷ DeltaToDivisor (E‑76)

where DpbOutputInterval[ n ] is specified in Equation C‑16 and DeltaToDivisor is specified in Table E.7 based on the value of frame\_field\_info\_present\_flag and pic\_struct for the CVS containing picture n. Entries marked "-" in Table E.7 indicate a lack of dependence of DeltaToDivisor on the corresponding syntax element.

When HighestTid is equal to i and ~~fixed\_pic\_rate\_general\_flag~~ fixed\_pic\_rate\_within\_cvs\_flag[ i ] is equal to 1 for a CVS containing picture n, the value computed for DpbOutputElementalInterval[ n ] shall be equal to ClockTick \* ( elemental\_duration\_in\_tc\_‌minus1[ i ] + 1 ), wherein ClockTick is as specified in Equation C‑1 (using the value of ClockTick for the CVS containing picture n) when one of the following conditions is true for the following picture in output order nextPicInOutputOrder that is specified for use in Equation C‑16:

– picture nextPicInOutputOrder is in the same CVS as picture n.

– picture nextPicInOutputOrder is in a different CVS and fixed\_pic\_rate\_general\_flag[ i ] is equal to 1 in the CVS containing picture nextPicInOutputOrder, the value of ClockTick is the same for both CVSs and the value of elemental\_duration\_in\_tc\_minus1[ i ] is the same for both CVSs.

~~When HighestTid is equal to i and fixed\_pic\_rate\_within\_cvs\_flag[ i ] is equal to 1 for a CVS containing picture n, the value computed for DpbOutputElementalInterval[ n ] shall be equal to ClockTick \* ( elemental\_duration\_in\_tc\_‌minus1[ i ] + 1 ), wherein ClockTick is as specified in Equation C‑1 (using the value of ClockTick for the CVS containing picture n) when the following picture in output order nextPicInOutputOrder that is specified for use in Equation C‑16 is in the same CVS as picture n.~~

**Table E.7 – Divisor for computation of DpbOutputElementalInterval[ n ]**

|  |  |  |
| --- | --- | --- |
| **frame\_field\_info\_present\_flag** | **pic\_struct** | **DeltaToDivisor** |
| 0 | - | 1 |
| 1 | 1 | 1 |
| 1 | 2 | 1 |
| 1 | 0 | 1 |
| 1 | 3 | 2 |
| 1 | 4 | 2 |
| 1 | 5 | 3 |
| 1 | 6 | 3 |
| 1 | 7 | 2 |
| 1 | 8 | 3 |
| 1 | 9 | 1 |
| 1 | 10 | 1 |
| 1 | 11 | 1 |
| 1 | 12 | 1 |

...

*In subclause F.7.4.3.1, make the following changes:*

...

**vps\_extension2\_flag** equal to 0 specifies that no vps\_extension\_data\_flag syntax elements are present in the VPS RBSP syntax structure. vps\_extension2\_flag equal to 1 specifies that vps\_extension\_data\_flag syntax elements ~~are~~ may be present in the VPS RBSP syntax structure. Decoders conforming to a profile specified in Annexes A, G or H shall ignore all data that follow the value 1 for vps\_extension2\_flag in a VPS RBSP.

*In subclause F.8.1.7, make the following changes:*

...

When this process is invoked, the following applies:

– For each RefPicSetStCurrBefore[ i ], with i in the range of 0 to NumPocStCurrBefore − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in clause 8.3.3.2 and the following applies:

– The value of PicOrderCntVal for the generated picture is set equal to PocStCurrBefore[ i ].

– The value of PicOutputFlag for the generated picture is set equal to 0.

– The generated picture is marked as "used for short-term reference".

– RefPicSetStCurrBefore[ i ] is set to be the generated reference picture.

– The value of nuh\_layer\_id for the generated picture is set equal to nuh\_layer\_id.

– The value of TemporalId for the generated picture is set equal to TemporalId of the current picture.

– The value of slice\_pic\_parameter\_set\_id for the generated picture is set equal to slice\_pic\_parameter\_set\_id of the current picture.

– For each RefPicSetStCurrAfter[ i ], with i in the range of 0 to NumPocStCurrAfter − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in clause 8.3.3.2 and the following applies:

– The value of PicOrderCntVal for the generated picture is set equal to PocStCurrAfter[ i ].

– The value of PicOutputFlag for the generated picture is set equal to 0.

– The generated picture is marked as "used for short-term reference".

– RefPicSetStCurrAfter[ i ] is set to be the generated reference picture.

– The value of nuh\_layer\_id for the generated picture is set equal to nuh\_layer\_id.

– The value of TemporalId for the generated picture is set equal to TemporalId of the current picture.

– The value of slice\_pic\_parameter\_set\_id for the generated picture is set equal to slice\_pic\_parameter\_set\_id of the current picture.

– For each RefPicSetStFoll[ i ], with i in the range of 0 to NumPocStFoll − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in clause 8.3.3.2 and the following applies:

– The value of PicOrderCntVal for the generated picture is set equal to PocStFoll[ i ].

– The value of PicOutputFlag for the generated picture is set equal to 0.

– The generated picture is marked as "used for short-term reference".

– RefPicSetStFoll[ i ] is set to be the generated reference picture.

– The value of nuh\_layer\_id for the generated picture is set equal to nuh\_layer\_id.

– The value of TemporalId for the generated picture is set equal to TemporalId of the current picture.

– The value of slice\_pic\_parameter\_set\_id for the generated picture is set equal to slice\_pic\_parameter\_set\_id of the current picture.

– For each RefPicSetLtCurr[ i ], with i in the range of 0 to NumPocLtCurr − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in clause 8.3.3.2 and the following applies:

– The value of PicOrderCntVal for the generated picture is set equal to PocLtCurr[ i ].

– The value of slice\_pic\_order\_cnt\_lsb for the generated picture is inferred to be equal to ( PocLtCurr[ i ] & ( MaxPicOrderCntLsb − 1 ) ).

– The value of PicOutputFlag for the generated picture is set equal to 0.

– The generated picture is marked as "used for long-term reference".

– RefPicSetLtCurr[ i ] is set to be the generated reference picture.

– The value of nuh\_layer\_id for the generated picture is set equal to nuh\_layer\_id.

– The value of TemporalId for the generated picture is set equal to TemporalId of the current picture.

– The value of slice\_pic\_parameter\_set\_id for the generated picture is set equal to slice\_pic\_parameter\_set\_id of the current picture.

– For each RefPicSetLtFoll[ i ], with i in the range of 0 to NumPocLtFoll − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in clause 8.3.3.2 and the following applies:

– The value of PicOrderCntVal for the generated picture is set equal to PocLtFoll[ i ].

– The value of slice\_pic\_order\_cnt\_lsb for the generated picture is inferred to be equal to ( PocLtFoll[ i ] & ( MaxPicOrderCntLsb − 1 ) ).

– The value of PicOutputFlag for the generated picture is set equal to 0.

– The generated picture is marked as "used for long-term reference".

– RefPicSetLtFoll[ i ] is set to be the generated reference picture.

– The value of nuh\_layer\_id for the generated picture is set equal to nuh\_layer\_id.

– The value of TemporalId for the generated picture is set equal to TemporalId of the current picture.

– The value of slice\_pic\_parameter\_set\_id for the generated picture is set equal to slice\_pic\_parameter\_set\_id of the current picture.

...

*In subclause F.13.2.3, make the following changes:*

The variables InitCpbRemovalDelay[ SchedSelIdx ] and InitCpbRemovalDelayOffset[ SchedSelIdx ] are updated, and the variables CpbDelayOffset and DpbDelayOffset are derived, as follows:

...

if( !concatenationFlag ) {  
 baseTime = AuNominalRemovalTime[ firstPicInPrevBuffPeriod ]  
 tmpCpbRemovalDelay = AuCpbRemovalDelayVal  
 tmpCpbDelayOffset = CpbDelayOffset  
} else {  
 baseTime1 = AuNominalRemovalTime[ prevNonDiscardablePic ]  
 ~~tmpCpbRemovalDelay =  
 Max( ( auCpbRemovalDelayDeltaMinus1 + 1 ),  
 Ceil( ( InitCpbRemovalDelay[ SchedSelIdx ] ÷ 90 000 +  
 AuFinalArrivalTime[ n − 1 ] − AuNominalRemovalTime[ n − 1 ] ) ÷ ClockTick ) )~~ tmpCpbRemovalDelay1 = ( auCpbRemovalDelayDeltaMinus1 + 1 )  
 baseTime2 = AuNominalRemovalTime[ n − 1 ]  
 tmpCpbRemovalDelay2 =   
 Ceil( ( InitCpbRemovalDelay[ SchedSelIdx ] ÷ 90000 +  
 AuFinalArrivalTime[ n − 1 ] − AuNominalRemovalTime[ n − 1 ] ) ÷ ClockTick ) (F‑76)  
 if( baseTime1 + ClockTick \* tmpCpbRemovalDelay1 <   
 baseTime2 + ClockTick \* tmpCpbRemovalDelay2 ) {  
 baseTime = baseTime2  
 tmpCpbRemovalDelay = tmpCpbRemovalDelay2  
 } else {  
 baseTime = baseTime1  
 tmpCpbRemovalDelay = tmpCpbRemovalDelay1  
 }  
 tmpCpbDelayOffset = 0  
}  
AuNominalRemovalTime[ n ] = baseTime + ClockTick \* ( tmpCpbRemovalDelay − tmpCpbDelayOffset )

...

*In subclause F.14.3.8, make the following changes:*

The alpha channel information SEI message provides information about alpha channel sample values and post-processing applied, an alpha blending process, to the decoded alpha planes coded in auxiliary pictures of type AUX\_ALPHA and one or more associated primary pictures. The alpha blending process is a process not specified by this Specification, in which an auxiliary coded picture is used in combination with a primary coded picture and with other data not specified by this Specification in the display process. In an alpha blending process, the samples of an auxiliary coded picture are interpreted as indications of the degree of opacity (or, equivalently, the degrees of transparency) associated with the corresponding luma samples of the primary coded picture.

...

**alpha\_channel\_cancel\_flag** equal to 1 indicates that the ~~alpha channel information~~ SEI message cancels the persistence of any previous alpha channel information SEI message in output order that applies to the current layer. alpha\_channel\_cancel\_flag equal to 0 indicates that alpha channel information follows.

...

**alpha\_channel\_incr\_flag** equal to 0 indicates that the interpretation sample value for each decoded auxiliary picture luma sample value is equal to the decoded auxiliary picture sample value for purposes of alpha blending. alpha\_channel\_incr\_flag equal to 1 indicates that, for purposes of alpha blending, after decoding the auxiliary picture samples, any auxiliary picture luma sample value that is greater than Min( alpha\_opaque\_value, alpha\_transparent\_value ) should be increased by one to obtain the interpretation sample value for the auxiliary picture sample and any auxiliary picture luma sample value that is less than or equal to Min( alpha\_opaque\_value, alpha\_transparent\_value ) should be used, without alteration, as the interpretation sample value for the decoded auxiliary picture sample value. ~~When not present, the value of alpha\_channel\_incr\_flag is inferred to be equal to 0.~~

When alpha\_channel\_incr\_flag is equal to 1, alpha\_transparent\_value shall not be equal to alpha\_opaque\_value and Log2( Abs( alpha\_opaque\_value − alpha\_transparent\_value ) ) shall have an integer value. A value of alpha\_transparent\_value that is equal to alpha\_opaque\_value indicates that the auxiliary coded picture is not intended for alpha blending purposes.

NOTE 1 – For alpha blending purposes, alpha\_opaque\_value may be greater than alpha\_transparent\_value, or it may be less than alpha\_transparent\_value. Interpretation sample values should be clipped to the range of alpha\_opaque\_value to alpha\_transparent\_value, inclusive.

**alpha\_channel\_clip\_flag** equal to 0 indicates that no clipping operation is applied to obtain the interpretation sample values of the decoded auxiliary picture. alpha\_channel\_clip\_flag equal to 1 indicates that the interpretation sample values of the decoded auxiliary picture are altered according to the clipping process described by the alpha\_channel\_clip\_type\_flag syntax element. When not present, the value of alpha\_channel\_clip\_flag is inferred to be equal to 0.

**alpha\_channel\_clip\_type\_flag** equal to 0 indicates that, for purposes of alpha blending, after decoding the auxiliary picture samples, any auxiliary picture luma sample that is greater than ~~( alpha\_opaque\_value − alpha\_transparent\_value ) / 2~~ ( alpha\_opaque\_value + alpha\_transparent\_value ) / 2 is set equal to alpha\_opaque\_value to obtain the interpretation sample value for the auxiliary picture luma sample and any auxiliary picture luma sample that is less or equal than ~~( alpha\_opaque\_value − alpha\_transparent\_value ) / 2~~ ( alpha\_opaque\_value + alpha\_transparent\_value ) / 2 is set equal to alpha\_transparent\_value to obtain the interpretation sample value for the auxiliary picture luma sample. alpha\_channel\_clip\_type\_flag equal to 1 indicates that, for purposes of alpha blending, after decoding the auxiliary picture samples, any auxiliary picture luma sample that is greater than alpha\_opaque\_value is set equal to alpha\_opaque\_value to obtain the interpretation sample value for the auxiliary picture luma sample and any auxiliary picture luma sample that is less than or equal to alpha\_transparent\_value is set equal to alpha\_transparent\_value to obtain the interpretation sample value for the auxiliary picture luma sample.

NOTE ~~1~~2 – When both alpha\_channel\_incr\_flag and alpha\_channel\_clip\_flag are equal to one, the clipping operation specified by alpha\_channel\_clip\_type\_flag should be applied first followed by the alteration specified by alpha\_channel\_incr\_flag to obtain the interpretation sample value for the auxiliary picture luma sample.

NOTE 3 – Alpha blending composition is normally performed with a background picture B, a foreground picture F, and a decoded auxiliary coded picture A, all of the same size. Assume for purposes of example illustration that the chroma resolution of B and F have been upsampled to the same resolution as the luma. Denote corresponding samples of B, F and A by b, f and a, respectively. Denote luma and chroma samples by subscripts Y, Cb and Cr.

Define the variables alphaRange, alphaFwt and alphaBwt as follows:

alphaRange = Abs( alpha\_opaque\_value − alpha\_transparent\_value ) (F‑95)

alphaFwt = Abs( a − alpha\_transparent\_value ) (F‑96)

alphaBwt = Abs( a − alpha\_opaque\_value ) (F‑97)

Then, in alpha blending composition, samples d of the displayed picture D could be calculated as follows:

dY = ( alphaFwt \* fY + alphaBwt \* bY + alphaRange / 2 ) / alphaRange (F‑98)

dCb = ( alphaFwt \* fCb + alphaBwt \* bCb + alphaRange / 2 ) / alphaRange (F‑99)

dCr = ( alphaFwt \* fCr + alphaBwt \* bCr + alphaRange / 2 ) / alphaRange (F‑100)

The samples of pictures D, F and B could also represent red, green, and blue component values (see clause 7.3). Here the Y, Cb and Cr component values are assumed. Each component, e.g., Y, is assumed for purposes of example illustration above to have the same bit depth in each of the pictures D, F and B. However, different components, e.g., Y and Cb, need not have the same bit depth in this example.

If alpha\_channel\_use\_idc is equal to 0, the following applies:

* F would be the decoded picture obtained from the decoded luma and chroma, and A would be the decoded picture obtained from the decoded auxiliary coded picture. In this case, the indicated example alpha blending composition involves multiplying the samples of F by factors obtained from the samples of A.
* A picture format that is useful for editing or direct viewing, and that is commonly used, is called pre-multiplied-black video. When the foreground picture was F, the pre-multiplied-black video S is given by

sY = ( alphaFwt \* fY ) / alphaRange (F‑101)

sCb = ( alphaFwt \* fCb ) / alphaRange (F‑102)

sCr = ( alphaFwt \* fCr ) / alphaRange (F‑103)

* Pre-multiplied-black video has the characteristic that the picture S will appear correct if displayed against a black background. For a non-black background B, the composition of the displayed picture D may be calculated as

dY = sY + ( alphaBwt \* bY + alphaRange / 2 ) / alphaRange (F‑104)

dCb = sCb + ( alphaBwt \* bCb + alphaRange / 2 ) / alphaRange (F‑105)

dCr = sCr + ( alphaBwt \* bCr + alphaRange / 2 ) / alphaRange (F‑106)

Otherwise, when alpha\_channel\_use\_idc is equal to 1, alpha blending composition does not involve multiplication of the samples of S by factors obtained from the samples of A, S would be the decoded picture obtained from the decoded luma and chroma, and A would again be the decoded picture obtained from the decoded auxiliary coded picture.

*In subclause F.14.3.9 and afterwards in Annex F, renumber all the equation indices.*

*In subclause G.14.2.4.1, make the following changes:*

|  |  |
| --- | --- |
| depth\_representation\_info( payloadSize ) { | **Descriptor** |
| **...** |  |
| **depth\_nonlinear\_representation\_model**[ i ] | ue(v) |
| **...** |  |
| } |  |

*In subclause G.14.3.4.1, make the following changes:*

...

**depth\_representation\_type** specifies the representation definition of decoded luma samples of auxiliary pictures as specified in Table G.3. In Table G.3, disparity specifies the horizontal displacement between two texture views and Z value specifies the distance from a camera.

The variable maxVal is set equal to ( 1  <<  ( 8 + bit\_depth\_luma\_minus8 ) ) − 1, where bit\_depth\_luma\_minus8 is the value included in or inferred for the active SPS of the layer with nuh\_layer\_id equal to targetLayerId. The value of depth\_representation\_type shall be in the range of 0 to 3, inclusive, in bitstreams conforming to this version of this Specification. The values of 4 to 15, inclusive, for depth\_representation\_type are reserved for future use by ITU-T | ISO/IEC. Although the value of depth\_representation\_type is required to be in the range of 0 to 3, inclusive, in this version of this Specification, decoders shall allow values of depth\_representation\_type in the range of 4 to 15, inclusive, to appear in the syntax. Decoders conforming to this version of this Specification shall ignore all data that follow a value of depth\_representation\_type in the range of 4 to 15, inclusive, in the depth representation information SEI message.

**Table G.3 – Definition of depth\_representation\_type**

|  |  |
| --- | --- |
| **depth\_representation\_type** | **Interpretation** |
| 0 | Each decoded luma sample value of an auxiliary picture represents an inverse of Z value that is uniformly quantized into the range of 0 to maxVal, inclusive.  When z\_far\_flag is equal to 1, the luma sample value equal to 0 represents the inverse of ZFar (specified below). When z\_near\_flag is equal to 1, the luma sample value equal to maxVal represents the inverse of ZNear (specified below). |
| 1 | Each decoded luma sample value of an auxiliary picture represents disparity that is uniformly quantized into the range of 0 to maxVal, inclusive.  When d\_min\_flag is equal to 1, the luma sample value equal to 0 represents DMin (specified below). When d\_max\_flag is equal to 1, the luma sample value equal to maxVal represents DMax (specified below). |
| 2 | Each decoded luma sample value of an auxiliary picture represents a Z value uniformly quantized into the range of 0 to maxVal, inclusive.  When z\_far\_flag is equal to 1, the luma sample value equal to 0 corresponds to ZFar (specified below). When z\_near\_flag is equal to 1, the luma sample value equal to maxVal represents ZNear (specified below). |
| 3 | Each decoded luma sample value of an auxiliary picture represents a non-linearly mapped disparity, normalized in range from 0 to maxVal, as specified by depth\_nonlinear\_representation\_num\_minus1 and depth\_nonlinear\_representation\_model[ i ].  When d\_min\_flag is equal to 1, the luma sample value equal to 0 represents DMin (specified below). When d\_max\_flag is equal to 1, the luma sample value equal to maxVal represents DMax (specified below). |
| ~~Other values~~ 4..15 | Reserved ~~for future use~~ |

**disparity\_ref\_view\_id** specifies the ViewId value ~~against~~ for which the disparity values are derived. The value of disparity\_ref\_view\_id shall be in the range of 0 to 1023, inclusive.

NOTE 1 – disparity\_ref\_view\_id is present only if d\_min\_flag is equal to 1 or d\_max\_flag is equal to 1 and is useful for depth\_representation\_type values equal to 1 and 3.

The variables in the x column of Table G.4 are derived from the respective variables in the s, e, n and v columns of Table G.4 as follows:

– If the value of e is in the range of 0 to 127, exclusive, x is set equal to ( −1 )s \* 2e − 31 \* ( 1 + n ÷ 2v ).

– Otherwise (e is equal to 0), x is set equal to ( −1 )s \* 2−( 30 + v ) \* n.

NOTE ~~1~~2 – The above specification is similar to that found in IEC 60559:1989.

**Table G.4 – Association between depth parameter variables and syntax elements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **x** | **S** | **e** | **n** | **v** |
| ZNear | ZNearSign | ZNearExp | ZNearMantissa | ZNearManLen |
| ZFar | ZFarSign | ZFarExp | ZFarMantissa | ZFarManLen |
| DMax | DMaxSign | DMaxExp | DMaxMantissa | DMaxManLen |
| DMin | DMinSign | DMinExp | DMinMantissa | DMinManLen |

The DMin and DMax values, when present, are specified in units of a luma sample width of the coded picture with ViewId equal to ViewId of the auxiliary picture.

The units for the ZNear and ZFar values, when present, are identical but unspecified.

**depth\_nonlinear\_representation\_num\_minus1** plus 2 specifies the number of piece-wise linear segments for mapping of depth values to a scale that is uniformly quantized in terms of disparity. The value of depth\_nonlinear\_representation\_num\_minus1 shall be in the range of 0 to 62, inclusive.

**depth\_nonlinear\_representation\_model**[ i ] for i ranging from 0 to depth\_nonlinear\_representation\_num\_minus1 + 2, inclusive, specify the piece-wise linear segments for mapping of decoded luma sample values of an auxiliary picture to a scale that is uniformly quantized in terms of disparity. The value of depth\_nonlinear\_representation\_model[ i ] shall be in the range of 0 to 65 535, inclusive. The values of depth\_nonlinear\_representation\_model[ 0 ] and depth\_nonlinear\_representation\_model[ depth\_nonlinear\_representation\_num\_minus1 + 2 ] are both inferred to be equal to 0.

NOTE ~~2~~3 – When depth\_representation\_type is equal to 3, an auxiliary picture contains non-linearly transformed depth samples. The variable DepthLUT[ i ], as specified below, is used to transform decoded depth sample values from the non-linear representation to the linear representation, i.e., uniformly quantized disparity values. The shape of this transform is defined by means of line-segment approximation in two-dimensional linear-disparity-to-non-linear-disparity space. The first ( 0, 0 ) and the last ( maxVal, maxVal ) nodes of the curve are predefined. Positions of additional nodes are transmitted in form of deviations (depth\_nonlinear\_representation\_model[ i ]) from the straight-line curve. These deviations are uniformly distributed along the whole range of 0 to maxVal, inclusive, with spacing depending on the value of nonlinear\_depth\_representation\_num\_minus1.

The variable DepthLUT[ i ] for i in the range of 0 to maxVal, inclusive, is specified as follows:

for( k = 0; k <= depth\_nonlinear\_representation\_num\_minus1 + 1; k++ ) {  
 pos1 = ( maxVal \* k ) / (depth\_nonlinear\_representation\_num\_minus1 + 2 )  
 dev1 = depth\_nonlinear\_representation\_model[ k ]  
 pos2 = ( maxVal \* ( k + 1 ) ) / (depth\_nonlinear\_representation\_num\_minus1 + 2 )  
 dev2 = depth\_nonlinear\_representation\_model[ k + 1 ] (G‑10)  
  
 x1 = pos1 − dev1  
 y1 = pos1 + dev1  
 x2 = pos2 − dev2  
 y2 = pos2 + dev2  
  
 for( x = Max( x1, 0 ); x <= Min( x2, maxVal ); x++ )  
 DepthLUT[ x ] = Clip3( 0, maxVal, Round( ( ( x − x1 ) \* ( y2 − y1 ) ) ÷ ( x2 − x1 ) + y1 ) )  
}

When depth\_representation\_type is equal to 3, DepthLUT[ dS ] for all decoded luma sample values dS of an auxiliary picture in the range of 0 to maxVal, inclusive, represents disparity that is uniformly quantized into the range of 0 to maxVal, inclusive.

*In subclause G.14.3.4.1, make the following changes:*

...

The depth\_rep\_info\_element( OutSign, OutExp, OutMantissa, OutManLen ) syntax structure sets the values of the OutSign, OutExp, OutMantissa and OutManLen variables that represent a floating-point value. When the syntax structure is included in another syntax structure, the variable names OutSign, OutExp, OutMantissa and OutManLen are to be interpreted as being replaced by the variable names used when the syntax structure is included.

**da\_sign\_flag** equal to 0 indicates that the sign of the floating-point value is positive. da\_sign\_flag equal to 1 indicates that the sign is negative. The variable OutSign is set equal to da\_sign\_flag.

**da\_exponent** specifies the exponent of the floating-point value. The value of da\_exponent shall be in the range of 0 to 27 − 2, inclusive. The value 27 − 1 is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 27 − 1 as indicating an unspecified value. The variable OutExp is set equal to da\_exponent.

**da\_mantissa\_len\_minus1** plus 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 subclause I.8.5.3.1, make the following changes:*

...

The decoding process for prediction units in inter prediction mode consists of the following ordered steps:

* 1. The derivation process for motion vector components and reference indices as specified in clause I.8.5.3.2 is invoked with the luma coding block location ( xCb, yCb ), the luma prediction block location ( xBl, yBl ), the luma coding block size block nCbS, the luma prediction block width nPbW, the luma prediction block height nPbH, and the prediction unit index partIdx as inputs, and the luma motion vectors mvL0 and mvL1, when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1, the reference indices refIdxL0 and refIdxL1, the prediction list utilization flags predFlagL0 and predFlagL1, and the flag subPbMotionFlag as outputs.
  2. Depending on the value of subPbMotionFlag and DbbpFlag[ xCb ][ yCb ], the following applies:

If both subPbMotionFlag and DbbpFlag[ xCb ][ yCb ] are equal to 0, the decoding process for inter sample prediction as specified in clause I.8.5.3.3.1 is invoked with the luma coding block location ( xCb, yCb ), the luma prediction block location ( xBl, yBl ), the luma coding block size block nCbS, the luma prediction block width nPbW, the luma prediction block height nPbH, the luma motion vectors mvL0 and mvL1, when ChromaArrayType is not equal to 0, the chroma motion vectors mvCL0 and mvCL1, the reference indices refIdxL0 and refIdxL1, and the prediction list utilization flags predFlagL0 and predFlagL1 as inputs, and the inter prediction samples (predSamples) that are an (nCbSL)x(nCbSL) array predSamplesL of prediction luma samples and, when ChromaArrayType is not equal to 0, two (nCbSwC)x(nCbShC) arrays ~~predSamples~~~~Cr~~predSamplesCb and predSamplesCr of prediction chroma samples, one for each of the chroma components Cb and Cr, as outputs.

...

*Throughout the document, replace all the eight instances of "decoder conformance to profiles" with "the decoding process". These instances are as follows:*

**vps\_extension\_data\_flag** may have any value. Its presence and value do not affect decoder conformance to profiles specified in Annex A. Decoders conforming to a profile specified in Annex A but not supporting the INBLD capability specified in Annex F shall ignore all vps\_extension\_data\_flag syntax elements.

**sps\_extension\_data\_flag** may have any value. Its presence and value do not affect decoder conformance to profiles specified in this version of this Specification. Decoders conforming to this version of this Specification shall ignore all sps\_extension\_data\_flag syntax elements.

**pps\_extension\_data\_flag** may have any value. Its presence and value do not affect decoder conformance to profiles specified in this version of this Specification. Decoders conforming to this version of this Specification shall ignore all pps\_extension\_data\_flag syntax elements.

**slice\_segment\_header\_extension\_data\_byte** may have any value. Decoders shall ignore the value of slice\_segment\_header\_extension\_data\_byte. Its value does not affect decoder conformance to profiles specified in Annex A.

**vps\_extension\_data\_flag** may have any value. Its presence and value do not affect decoder conformance to profiles specified in Annexes A, G or H. Decoders conforming to a profile specified in Annexes A, G or H shall ignore all vps\_extension\_data\_flag syntax elements.

**vps\_non\_vui\_extension\_data\_byte** may have any value. Decoders shall ignore the value of vps\_non\_vui\_extension\_data\_byte. Its value does not affect decoder conformance to profiles specified in this version of this Specification.

**slice\_segment\_header\_extension\_data\_bit** may have any value. Decoders shall ignore the value of slice\_segment\_header\_extension\_data\_bit. Its value does not affect decoder conformance to profiles specified in this version of this Specification.

**vps\_extension\_data\_flag** may have any value. Its presence and value do not affect decoder conformance to profiles specified in Annexes A, G, H, or I. Decoders conforming to a profile specified in Annexes A, G, H, or I shall ignore all vps\_extension\_data\_flag syntax elements.

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