Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11
5th Meeting: Vienna, AT, 27 Jul. –2 Aug. 2013

Document: JCT3V-E0100_spec_v1

Title: **JCT-3V AHG 3: Editorial improvements on MV-HEVC Draft Text 4**

Status: Input document

Purpose: Proposal

Author(s) or Contact(s):

Gerhard Tech
Fraunhofer HHI
Email: gerhard.tech@hhi.fraunhofer.de

Krzysztof Wegner
Poznan University of Technology
Email: kwegner@multimedia.edu.pl

Ying Chen
Qualcomm Incorporated
Email: cheny@qti.qualcomm.com

Miska Hannuksela
Nokia Corporation
Email: miska.hannuksela@nokia.com

Jill Boyce
Vidyo
Email: jill@vidyo.com

Source: Editors

ABSTRACT

Modifications in long sections copied from the HEVC spec are highlighted in turquoise. Open issues and editor's notes are highlighted in yellow.

Ed. Notes (E0100)

- ---------- Release v1 ----------
- Accepted all changes in document.
- ---------- v0 ----------
- (Cleanup GT01) Fixed references and scope of restructered Annexes.
- (Restructured Annexes) Moved clauses from Annex G to Annex F
- (Incorporated General SEI message syntax)
- (Changed semantics order in slice header) to match syntax table.
- (Review GT01) Review, typo corrections, editorial improvement, clean ups.
- (Moved RPS decoding process) Removed RPS decoding process for reference picture set of the same layer and added changes to base spec.

Ed. Notes (Draft 4)

- ---------- Release v4 ----------
- Accepted all change marks.
- ---------- Release v3 (d2) ----------
- (Version numbering) Changed document numbering from zero-based to one-based. (_d2 becomes -v3)
- (Review GT08) Review, typo corrections, editorial improvement, clean ups.
- (Review GT07) Review, typo corrections, editorial improvement, clean ups.
- (Update note 3D Display SEI) Updated note in 3D reference display SEI based on new text provided of the proponent.
- (Review GT06) Review, typo corrections, editorial improvement, clean ups.
- (M0457): Bug fix to use the information indicated through the inter-layer reference picture set in alt_collocated syntax elements rather than the VPS information directly. Previously, the semantics of inter_layer_pred_enabled, num_inter_layer_ref_pics_minus1 and inter_layer_pred_layer[i] concerned all types of inter-layer prediction but actually only affected sample prediction and it was possible to use motion prediction from a reference layer not listed in inter_layer_pred_layer_idc[i]. Now the inter-layer motion prediction is also constrained to take place among layers indicated by inter_layer_pred_layer_idc[i].
- (Review MH02): Review, clean ups, editor's notes.
- (Review JB02) Review, clean ups, add editors notes, definitions, and missing contstraint
- (Joint/M0264 and M0208) AU definition and other editorial improvements
- (Review GT05) Review, typo corrections, editorial improvement, clean ups.
- (SHVC, Reserved) Changed SHVC syntax and semantics to reserved values (To be discussed).
- (SF/M0040/single_layer_for_non_irap_flag) Adaptive resolution change and efficient trick
- (PP/M0463/Parallel processing delay indication) Incorporated improved version provided by the editors.
- (3D/D0220/ViewId) Adopt view id aspect.
- ---------- Release d1 ----------
- (PP/M0464/Tile alignment flag) Adopt first aspect (tile boundary alignment flag). Editorial improvement needed.
- (Copied text from HEVC version 1) VUI related. .
- (Removed AltCollocatedIndicationFlag)
- (SF/M0309/Extended spatial scalability) Signalling of extended spatial scalability.
- (Review JB01) Move direct_dependency_flag semantics to correct location, improvements to M0458.
- (PS/M0268/Output Layer Sets, Profile Tier) #7 Section 3 of the v2 document; An alternative method for signalling of profile, tier, and level information and output layer sets
- (Added stereo main profile scalable restriction) as suggested by Miska for consideration.
- (Added inference SPS syntax elements) for sps_max_sub_layers_minus1 sps_temporal_id_nesting_flag.
- (Copied text from HEVC version 1) sps_max_sub_layers_minus1 sps_temporal_id_nesting_flag.
- (Fix bit length for num_inter_layer_ref_pics_minus1)
- (Move if-statement in 8.1.1) after "When the current picture is an IRAP picture, the following applies:" to 8.1.
- (Renamed max_sublayer_for_ilp_plus1) to max_tid_il_ref_pics_plus1.
- (Renamed LayerIdInVps) to LayerIdxInVps.
- (Removal InterLayerMfmEnableFlag) and related notes. Should be incorporated in SHVC draft.
- (Review GT04) Review, typo corrections, editorial improvement, clean ups.
- ---------- Release d0 ----------
- (Removed old marking) Removed old spec text of Marking process for sub-layer non-reference pictures not needed for inter-layer prediction.

- (Removed LayerSetPresentFlag) Removed LayerSetPresent as discussed.
- (Review GT03) Review, typo corrections, editorial improvement, clean ups.
- (Review MH01) Review, typo corrections, editorial improvement, clean ups.
- (Review JB + YW) Review, typo corrections, editorial improvement, clean ups.
- (Review GT02) Review, typo corrections, editorial improvement, clean ups.
- (RPSM/M0458/Active inter-layer ref pics in slice header) #18 1.) max_one_active_ref_layer_flag in VPS, 2.) slice segment header indicates inter-layer ref. pics, 3.) Change IL-RPS and ref pic list construction. Have a semantic constraint that inter_layer_idc[i] shall be increasing. Editorial notes that further improvements related to aspect 3 are encouraged. Also agreed to let the editors to combine text of JCTVC-M0457 and JCTVC-M0458. Includes resolution of editorial issue identified under SILP/M0209/IL RPS decoding.
- (SILP/M0457/Dependency type, Alt coll. ref. idx., TMVP change) #16 Signalling of inter layer prediction type (motion/sample), alternative collocated picture, flags for kind of enabled inter-layer prediction per slice, modified TMVP)
- (PS/D0311/Dim. ID not when SplittingFlag) #9 Replaces a semantic constraint on dimension_id with an inference when splitting flag is equal to 1. Ed. improvement needed to handle setting default values for scalability type dimensions that are not present.
- (SEI/D0218/3DRefDispSEI) #23 3D reference displays information SEI message.
- (SEI/M0043/Layers present SEI message) #22 Agreed with the following change: the persistence scope of the SEI message should be further restricted to be within a CVS.
- (SILP/M0162/discardable_flag dependent marking) #15 A picture that has nuh_layer_id greater than 0 and discardable_flag equal to 1 is marked as "unused for reference" after its decoding.
- (SILP/M0152/discardable_flag) #14 One reserved flag in the slice header, when equal to 1, indicates that the picture is not used for inter-layer prediction and not used for inter prediction.
- (Ed. Add slice segment header) Added slice segment header syntax table from HEVC 1.
- (SILP/M0209/IL RPS decoding) #13 Decoding of inter-layer reference picture set and reference picture list construction based on TemporalId. An editorial improvement is needed regarding the deviation of a variable NumInterLayerRpsPics that is currently in the decoding process.
- (SILP/M0209/marking non ref temp sub layer) #12 Marking of certain pictures as "unused for reference" base on max_sublayer_for_ilp_plus1.
- (SILP/M0203/max_sublayer_for_ilp_plus1) #11 Signalling of maximum TemporalId used in inter-layer prediction.. Agreed with a change "<=" to "<" in the loop of the added syntax.
- (PS/M0163/No sig.last dimension_id_len_minus1) #10 No signalling of the last dimension_id_len_minus1[i], when splitting_flag is equal to 1.
- (PS/M0268/SPS Flag signalling) #8 Don't signal sps_max_sub_layers_minus1 and sps_temporal_id_nesting_flag when nuh_layer_id_> 0.
- (PS/M0268/output_layer_set_idx) #6 Change the syntax element output_layer_set_idx[i] to output_layer_set_idx_minus1[i].
- (PS/M0268/PositionDirectDependencyFlags) #5 Move the direct dependency flag syntax section to directly follow the dimension_id syntax (ahead of profile/tier/level) signalling.
- (Joint/M0208/NumPocTotalCurr) Clarify that the value of NumPocTotalCurr shall be equal to 0 for a BLA or CRA picture if nuh_layer_id is equal to 0.
- (Joint/M0045/Stereo Main/no mixed scal.) The principle not to support mixed scalability types for now. Concrete language to be worked.
- (Joint/M0168/AUD Layer Id) #1 The allowed layer ID value for the AUD should correspond to the lowest VCL NAL unit layer ID in the AU.
- (Joint/M0168/SPS activation) An IRAP NAL unit of each layer with NoRaslOutputFlag equal to 1 may activate a new SPS for the corresponding layer
- (Review GT01) Review, typo corrections, editorial improvement, clean ups.

Ed. Notes (Draft 3) (changes to JCT3V-B1004)

- ---------- Release d2 ----------
- (Review GT3) Editorial clean-ups.
- (Update 2 to latest HEVC spec) Update to JCTVC-L1003_v33.doc.
- (Fix SPS profile_tier_level) Fixed signalling of profile_tier_level syntax structure in SPS.
- (SPS to Annex F) Moved SPS syntax and semantics from Annex G to Annex F.
- (Ed. note on pic size) Added editor's note on picture size restriction.
- (Fix bit masking for splitting) Corrected of erroneous bit shift.
- ---------- Release d1 ----------
- (Review GT2)
- (Review MH2)
- (JCT3V-C0238 Marking Process) Replace targetDecLayerIdList by TargetDecLayerIdList in F.8.1.2.1.

- (Ed. Notes 01) Incorporated and removed editor's notes as discussed.
- (JCT3V-0059) Update to new terminology and simplification of text.
- (Fix References) Fix of references and numbering.
- (JCTVC-L0363) Fixed byte alignment corrupted when re-introduction of profilePresentFlag in profile_tier_level for JCTVC-L0180.
- (JCTVC-L0180) Updated of semantics and modified profile_tier_level syntax structure.
- (Review GT01)
- (JCT3V-C0078) Incorporated disparity vector constraints.
- (Update to latest HEVC spec), Updated to JCTVC-L1003_v19.doc.
- (Review Miska01)
- (MVC-CY01) Review, typo corrections, editorial improvement and alignment with B1004.
- (JCT3V-C0085) Integration of JCT3V-C0085: slice type constraint.
- ---------- Release d0 ----------
- (JCT3V-C0238) Incorporated common specification text for scalable multi-view extensions.

Ed. Notes (WD2) (based on JCT3V-A1004)

- ---------- Release d0 ----------
- (MVC-MH) Review, typo corrections, editorial improvement, and editor's notes
- (MVE-06) Incorporated introductory paragraph for view dependency change SEI message.
- (MVE-05) Incorporated invocation of sub-bitstream extraction process in general decoding process
- (MVE-04) Fixed construction of layerId list in general decoding process
- (MVC-KW) Review, typo corrections, editorial improvement.
- (MVE-03) Replacement of changes marks related to base spec by highlighting
- (MVE-01/JCT3V-B00046) Incorporated editorial note
- (MVC-GT) Review, typo corrections, editorial improvement.
- (MVC-CY) Review, typo corrections, editorial improvement.
- (MVE-02) Incorporated initial version of HRD text.
- (MVN-01/JCT3V-B0063) Incorporated view dependency change SEI.

Ed. Notes (WD1) (based on : JCT3V-A0012)

- ---------- Release d0 ----------
- (Rev3, KW) Review and small corrections
- (Rev2, GT) Review and text improvement
- Missing part in general decoding process
- (Replacement view_id by layer_id)
- Font issue fix.
- (Fix: picture marking)
- (Rev1, CY), Review and small corrections
- (MV07) Fix references
- (MV06) Improvement and update of interview prediction text.
- (MV02,MV03) Update of high level syntax and definitions
- (MV09) general HEVC decoding process
- (MV08) Additional sections/placeholders
- (MV04, MV05) Removal of low-level and depth tools
- (MV01) Removed HEVC spec
- Update of low level specification to match HEVC text specification 8(d7)

CONTENTS

Page

Abstract ii
Contents v
8 Decoding process 1
8.1 General decoding process 1
8.1.1 Decoding process for a coded picture with nuh_layer_id equal to 0 1
8.2 NAL unit decoding process 2
8.3 Slice decoding process 3
8.3.1 Decoding process for picture order count 3
8.3.2 Decoding process for reference picture set 3
Annex D Supplemental enhancement information 8
D.1 SEI payload syntax 8
D.1.1 General SEI message syntax 8
Annex F Common syntax, semantics and decoding processes for multi-layer video coding extensions 10
F.1 Scope 10
F.2 Normative references 10
F.3 Definitions 10
F.4 Abbreviations 11
F.5 Conventions 11
F.6 Source, coded, decoded and output data formats, scanning processes, and neighbouring relationships 11
F.7 Syntax and semantics 11
F.7.1 Method of specifying syntax in tabular form 11
F.7.2 Specification of syntax functions, categories, and descriptors 11
F.7.3 Syntax in tabular form 11
F.7.3.1 NAL unit syntax 11
F.7.3.2 Raw byte sequence payloads and RBSP trailing bits syntax 12
F.7.3.3 Profile, tier and level syntax 17
F.7.3.4 Scaling list data syntax 18
F.7.3.5 Supplemental enhancement information message syntax 18
F.7.3.6 Slice segment header syntax 18
F.7.3.7 Short-term reference picture set syntax 21
F.7.3.8 Slice segment data syntax 21
F.7.4 Semantics 22
F.7.4.1 General 22
F.7.4.2 NAL unit semantics 22
F.7.4.3 Raw byte sequence payloads, trailing bits, and byte alignment semantics 23
F.7.4.4 Profile, tier and level semantics 28
F.7.4.5 Scaling list data semantics 29
F.7.4.6 Supplemental enhancement information message semantics 29
F.7.4.7 Slice segment header semantics 29
F.7.4.8 Short-term reference picture set semantics 31
F.7.4.9 Slice segment data semantics 31
F.8 Decoding process 32
F.8.1 General decoding process 32
F.8.1.1 Decoding process for starting the decoding of a coded picture with nuh_layer_id greater than 0 32
F.8.1.2 Decoding process for ending the decoding of a coded picture with nuh_layer_id greater than 0 33
F.8.2 NAL unit decoding process 33
F.8.3 Slice decoding processes 34
F.8.3.1 (void) 34
F.8.3.2 (void) 34
F.8.3.3 (void) 34
F.8.3.4 Decoding process for reference picture lists construction 34
F.8.4 Decoding process for coding units coded in intra prediction mode 35
F.8.5 Decoding process for coding units coded in inter prediction mode 35
F.8.5.1 Derivation process for temporal luma motion vector prediction 35
F.8.6 Scaling, transformation and array construction process prior to deblocking filter process 36
F.8.7 In-loop filter process 36

F.9 Parsing process 36
F.10 Specification of bitstream subsets 36
F.11 (Void) 36
F.12 Byte stream format 36
F.13 Hypothetical reference decoder 36
F.13.1 General 36
F.13.2 Operation of coded picture buffer (CPB) 36
F.13.3 Operation of the decoded picture buffer (DPB) 37
F.13.3.1 General 37
F.13.3.2 Removal of pictures from the DPB 37
F.13.3.3 Picture output 37
F.13.3.4 Current decoded picture marking and storage 37
F.13.4 Bitstream conformance 37
F.13.5 Decoder conformance 39
F.13.5.1 General 39
F.13.5.2 Operation of the output order DPB 39
F.14 SEI messages 41
F.14.1 SEI message syntax 41
F.14.1.1 Layer dependency change SEI message syntax 41
F.14.1.2 Layers present SEI message syntax 41
F.14.2 SEI message semantics 41
F.14.2.1 Layer dependency change SEI message semantics 41
F.14.2.2 Layers present SEI message semantics 42
F.15 Video usability information 42
F.15.1 General 42
F.15.2 VUI syntax 42
F.15.2.1 VUI parameters syntax 43
F.15.2.2 HRD parameters syntax 44
F.15.2.3 Sub-layer HRD parameters syntax 44
F.15.3 VUI semantics 44
F.15.3.1 VUI parameters semantics 44
F.15.3.2 HRD parameters semantics 46
F.15.3.3 Sub-layer HRD parameters semantics 47
Annex G Multiview coding 48
G.1 Scope 48
G.2 Normative references 48
G.3 Definitions 48
G.4 Abbreviations 48
G.5 Conventions 48
G.6 Source, coded, decoded and output data formats, scanning processes, and neighbouring relationships 48
G.7 Syntax and semantics 48
G.8 Decoding processes 48
G.8.1 General decoding process 48
G.8.1.1 Decoding process for a coded picture with nuh_layer_id greater than 0 48
G.8.1.2 Decoding process for inter-layer reference picture set 49
G.8.1.3 Marking process for ending the decoding of a coded picture with nuh_layer_id greater than 0 49
G.8.2 NAL unit decoding process 49
G.8.3 Slice decoding processes 49
G.8.3.1 Decoding process for picture order count 49
G.8.3.2 Decoding process for reference picture set 49
G.8.3.3 Decoding process for generating unavailable reference pictures 49
G.8.3.4 Decoding process for reference picture lists construction 49
G.8.4 Decoding process for coding units coded in intra prediction mode 49
G.8.5 Decoding process for coding units coded in inter prediction mode 49
G.8.6 Scaling, transformation and array construction process prior to deblocking filter process 49
G.8.7 In-loop filter process 50
G.9 Parsing process 50
G.10 Specification of bitstream subsets 50
G.11 Profiles, tiers, and levels 50
G.11.1 Profiles 50
G.11.1.1 General 50

G.11.1.2 Stereo Main profile 50
G.11.2 Tiers and levels 50
G.12 Byte stream format 50
G.13 Hypothetical reference decoder 50
G.14 SEI messages 50
G.14.1 SEI message syntax 50
G.14.1.1 3D reference displays information SEI message syntax 50
G.14.2 SEI message semantics 51
G.14.2.1 3D reference displays information SEI message semantics 51
G.15 Video usability information 54

Replace or extend subclauses 8.1, 8.2 8.3 and D.2 with the following (with additions indicated in turquoise).

8 Decoding process

8.1 General decoding process

Input to this process is a bitstream. Output of this process is a list of decoded pictures.

The layer identifier list TargetDecLayerIdList, which specifies the list of nuh_layer_id values, in increasing order of nuh_layer_id values, of the NAL units to be decoded, is specified as follows:

– If some external means, not specified in this Specification, is available to set TargetDecLayerIdList, TargetDecLayerIdList is set by the external means.

– Otherwise, if the decoding process is invoked in a bitstream conformance test as specified in subclause C.1, TargetDecLayerIdList is set as specified in subclause C.1.

– Otherwise, TargetDecLayerIdList contains only one nuh_layer_id value that is equal to 0.

The variable HighestTid, which identifies the highest temporal sub-layer to be decoded, is specified as follows:

– If some external means, not specified in this Specification, is available to set HighestTid, HighestTid is set by the external means.

– Otherwise, if the decoding process is invoked in a bitstream conformance test as specified in subclause C.1, HighestTid is set as specified in subclause C.1.

– Otherwise, HighestTid is set equal to sps_max_sub_layers_minus1.

The sub-bitstream extraction process as specified in clause 10 is applied with the bitstream, HighestTid, and TargetDecLayerIdList as inputs, and the output is assigned to a bitstream referred to as BitstreamToDecode.

The decoding processes specified in the remainder of this subclause apply to each coded picture, referred to as the current picture and denoted by the variable CurrPic, in BitstreamToDecode.

Depending on the value of chroma_format_idc, the number of sample arrays of the current picture is as follows:

– If chroma_format_idc is equal to 0, the current picture consists of 1 sample array S_L.

– Otherwise (chroma_format_idc is not equal to 0), the current picture consists of 3 sample arrays S_L, S_{Cb}, S_{Cr}.

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

The decoding process is specified such that all decoders will produce numerically identical cropped decoded pictures. Any decoding process that produces identical cropped decoded pictures to those produced by the process described herein (with the correct output order or output timing, as specified) conforms to the decoding process requirements of this Specification.

When the current picture is an IRAP picture, the following applies:

– If the current picture with a particular nuh_layer_id is an IDR picture, a BLA picture, the first picture with that particular nuh_layer_id in the bitstream in decoding order, or the first picture with that particular nuh_layer_id that follows an end of sequence NAL unit in decoding order, the variable NoRaslOutputFlag is set equal to 1.

– Otherwise, if some external means not specified in this Specification is available to set the variable HandleCraAsBlaFlag to a value for the current picture, the variable HandleCraAsBlaFlag is set equal to the value provided by the external means and the variable NoRaslOutputFlag is set equal to HandleCraAsBlaFlag.

– Otherwise, the variable HandleCraAsBlaFlag is set equal to 0 and the variable NoRaslOutputFlag is set equal to 0.

When the current picture has nuh_layer_id equal to 0, the decoding process for a coded picture with nuh_layer_id equal to 0 specified in subclause 8.1.1 is invoked.

8.1.1 Decoding process for a coded picture with nuh_layer_id equal to 0

When the current picture is a BLA picture that has nal_unit_type equal to BLA_W_LP or is a CRA picture, the following applies:

– If some external means not specified in this Specification is available to set the variable UseAltCpbParamsFlag to a value, UseAltCpbParamsFlag is set equal to the value provided by the external means.

– Otherwise, the value of UseAltCpbParamsFlag is set equal to 0.

Depending on the value of separate_colour_plane_flag, the decoding process is structured as follows:

– If separate_colour_plane_flag is equal to 0, the decoding process is invoked a single time with the current picture being the output.

– Otherwise (separate_colour_plane_flag is equal to 1), the decoding process is invoked three times. Inputs to the decoding process are all NAL units of the coded picture with identical value of colour_plane_id. The decoding process of NAL units with a particular value of colour_plane_id is specified as if only a CVS with monochrome colour format with that particular value of colour_plane_id would be present in the bitstream. The output of each of the three decoding processes is assigned to one of the 3 sample arrays of the current picture, with the NAL units with colour_plane_id equal to 0, 1, and 2 being assigned to S_L, S_{Cb}, and S_{Cr}, respectively.

NOTE – The variable ChromaArrayType is derived as equal to 0 when separate_colour_plane_flag is equal to 1 and chroma_format_idc is equal to 3. In the decoding process, the value of this variable is evaluated resulting in operations identical to that of monochrome pictures (when chroma_format_idc is equal to 0).

The decoding process operates as follows for the current picture CurrPic:

1. The decoding of NAL units is specified in subclause 8.2.
2. The processes in subclause 8.3 specify the following decoding processes using syntax elements in the slice segment layer and above:
   – Variables and functions relating to picture order count are derived in subclause 8.3.1. This needs to be invoked only for the first slice segment of a picture.
   – The decoding process for RPS in subclause 8.3.2 is invoked, wherein reference pictures may be marked as "unused for reference" or "used for long-term reference". This needs to be invoked only for the first slice segment of a picture.
   – When the current picture is a BLA picture or is a CRA picture with NoRaslOutputFlag equal to 1, the decoding process for generating unavailable reference pictures specified in subclause 8.3.3 is invoked, which needs to be invoked only for the first slice segment of a picture.
   – PicOutputFlag is set as follows:
      – If the current picture is a RASL picture and NoRaslOutputFlag of the associated IRAP picture is equal to 1, PicOutputFlag is set equal to 0.
      – Otherwise, PicOutputFlag is set equal to pic_output_flag.
   – At the beginning of the decoding process for each P or B slice, the decoding process for reference picture lists construction specified in subclause 8.3.4 is invoked for derivation of reference picture list 0 (RefPicList0) and, when decoding a B slice, reference picture list 1 (RefPicList1).
3. The processes in subclauses 8.4, 8.5, 8.6, and 8.7 specify decoding processes using syntax elements in all syntax structure layers. It is a requirement of bitstream conformance that the coded slices of the picture shall contain slice segment data for every coding tree unit of the picture, such that the division of the picture into slices, the division of the slices into slice segments, and the division of the slice segments into coding tree units each form a partitioning of the picture.
4. After all slices of the current picture have been decoded, the decoded picture is marked as "used for short-term reference".

8.2 NAL unit decoding process

Inputs to this process are NAL units of the access unit containing the current picture. [Ed. (MH): This process should be modified to input NAL units of the coded picture and associated NAL units, as it is invoked separately for each coded picture within an access unit. For that, association of NAL units to coded pictures has to be specified. Alternatively, this process should be invoked only once per access unit, and the calling processes should be modified accordingly.]

Outputs of this process are the parsed RBSP syntax structures encapsulated within the NAL units of the access unit containing the current picture.

The decoding process for each NAL unit extracts the RBSP syntax structure from the NAL unit and then then parses the RBSP syntax structure.

8.3 Slice decoding process

8.3.1 Decoding process for picture order count

Output of this process is PicOrderCntVal, the picture order count of the current picture.

Picture order counts are used to identify pictures, for deriving motion parameters in merge mode and motion vector prediction, and for decoder conformance checking (see subclause C.5).

Each coded picture is associated with a picture order count variable, denoted as PicOrderCntVal.

When the current picture is not an IRAP picture with NoRaslOutputFlag equal to 1, the variables prevPicOrderCntLsb and prevPicOrderCntMsb are derived as follows:

– Let prevTid0Pic be the previous picture in decoding order that has TemporalId equal to 0 and nuh_layer_id equal to nuh_layer_id of the current picture and that is not a RASL picture, a RADL picture, or a sub-layer non-reference picture.

– The variable prevPicOrderCntLsb is set equal to slice_pic_order_cnt_lsb of prevTid0Pic.

– The variable prevPicOrderCntMsb is set equal to PicOrderCntMsb of prevTid0Pic.

The variable PicOrderCntMsb of the current picture is derived as follows:

– If the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, PicOrderCntMsb is set equal to 0.

– Otherwise, PicOrderCntMsb is derived as follows:

```
if( ( slice_pic_order_cnt_lsb <  prevPicOrderCntLsb )  &&
        ( ( prevPicOrderCntLsb − slice_pic_order_cnt_lsb )  >=  ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb + MaxPicOrderCntLsb                    (8-1)
else if( (slice_pic_order_cnt_lsb  >  prevPicOrderCntLsb )  &&
        ( (slice_pic_order_cnt_lsb − prevPicOrderCntLsb )  >  ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb − MaxPicOrderCntLsb
else
    PicOrderCntMsb = prevPicOrderCntMsb
```

PicOrderCntVal is derived as follows:

$$PicOrderCntVal = PicOrderCntMsb + slice\_pic\_order\_cnt\_lsb \quad (8\text{-}2)$$

NOTE 1 – All IDR pictures will have PicOrderCntVal equal to 0 since slice_pic_order_cnt_lsb is inferred to be 0 for IDR pictures and prevPicOrderCntLsb and prevPicOrderCntMsb are both set equal to 0.

The value of PicOrderCntVal shall be in the range of -2^{31} to $2^{31} - 1$, inclusive. In one CVS, the PicOrderCntVal values for any two coded pictures shall not be the same.

The function PicOrderCnt(picX) is specified as follows:

$$PicOrderCnt(picX) = PicOrderCntVal \text{ of the picture } picX \quad (8\text{-}3)$$

The function DiffPicOrderCnt(picA, picB) is specified as follows:

$$DiffPicOrderCnt(picA, picB) = PicOrderCnt(picA) - PicOrderCnt(picB) \quad (8\text{-}4)$$

The bitstream shall not contain data that result in values of DiffPicOrderCnt(picA, picB) used in the decoding process that are not in the range of -2^{15} to $2^{15} - 1$, inclusive.

NOTE 2 – Let X be the current picture and Y and Z be two other pictures in the same sequence, Y and Z are considered to be in the same output order direction from X when both DiffPicOrderCnt(X, Y) and DiffPicOrderCnt(X, Z) are positive or both are negative.

8.3.2 Decoding process for reference picture set

This process is invoked once per picture, after decoding of a slice header but prior to the decoding of any coding unit and prior to the decoding process for reference picture list construction for the slice as specified in subclause 8.3.3. This process may result in one or more reference pictures in the DPB being marked as "unused for reference" or "used for long-term reference". This subclause marks only the pictures with the same value of nuh_layer_id and does not mark any picture with a nuh_layer_id different from the current picture.

NOTE 1 – The RPS is an absolute description of the reference pictures used in the decoding process of the current and future coded pictures. The RPS signalling is explicit in the sense that all reference pictures included in the RPS are listed explicitly.

A decoded picture in the DPB can be marked as "unused for reference", "used for short-term reference", or "used for long-term reference", but only one among these three at any given moment during the operation of the decoding process. Assigning one of these markings to a picture implicitly removes another of these markings when applicable. When a picture is referred to as being marked as "used for reference", this collectively refers to the picture being marked as "used for short-term reference" or "used for long-term reference" (but not both).

When the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, all reference pictures currently in the DPB (if any) are marked as "unused for reference".

Short-term reference pictures are identified by their PicOrderCntVal values. Long-term reference pictures are identified either by their PicOrderCntVal values or their slice_pic_order_cnt_lsb values.

Five lists of picture order count values are constructed to derive the RPS. These five lists are PocStCurrBefore, PocStCurrAfter, PocStFoll, PocLtCurr, and PocLtFoll, with NumPocStCurrBefore, NumPocStCurrAfter, NumPocStFoll, NumPocLtCurr, and NumPocLtFoll number of elements, respectively. The five lists and the five variables are derived as follows:

– If the current picture is an IDR picture, PocStCurrBefore, PocStCurrAfter, PocStFoll, PocLtCurr, and PocLtFoll are all set to be empty, and NumPocStCurrBefore, NumPocStCurrAfter, NumPocStFoll, NumPocLtCurr, and NumPocLtFoll are all set equal to 0.

– Otherwise, the following applies:

```
for( i = 0, j = 0, k = 0; i < NumNegativePics[ CurrRpsIdx ] ; i++ )
    if( UsedByCurrPicS0[ CurrRpsIdx ][ i ] )
        PocStCurrBefore[ j++ ] = PicOrderCntVal + DeltaPocS0[ CurrRpsIdx ][ i ]
    else
        PocStFoll[ k++ ] = PicOrderCntVal + DeltaPocS0[ CurrRpsIdx ][ i ]
NumPocStCurrBefore = j

for( i = 0, j = 0; i < NumPositivePics[ CurrRpsIdx ]; i++ )
    if( UsedByCurrPicS1[ CurrRpsIdx ][ i ] )
        PocStCurrAfter[ j++ ] = PicOrderCntVal + DeltaPocS1[ CurrRpsIdx ][ i ]
    else
        PocStFoll[ k++ ] = PicOrderCntVal + DeltaPocS1[ CurrRpsIdx ][ i ]
NumPocStCurrAfter = j
NumPocStFoll = k                                                                    (8-5)
for( i = 0, j = 0, k = 0; i < num_long_term_sps + num_long_term_pics; i++ ) {
    pocLt = PocLsbLt[ i ]
    if( delta_poc_msb_present_flag[ i ] )
        pocLt  +=  PicOrderCntVal − DeltaPocMsbCycleLt[ i ] * MaxPicOrderCntLsb − slice_pic_order_cnt_lsb
    if( UsedByCurrPicLt[ i ] ) {
        PocLtCurr[ j ] = pocLt
        CurrDeltaPocMsbPresentFlag[ j++ ] = delta_poc_msb_present_flag[ i ]
    } else {
        PocLtFoll[ k ] = pocLt
        FollDeltaPocMsbPresentFlag[ k++ ] = delta_poc_msb_present_flag[ i ]
    }
}
NumPocLtCurr = j
NumPocLtFoll = k
```

where PicOrderCntVal is the picture order count of the current picture as specified in subclause 8.3.1.

NOTE 2 – A value of CurrRpsIdx in the range of 0 to num_short_term_ref_pic_sets − 1, inclusive, indicates that a candidate short-term RPS from the active SPS is being used, where CurrRpsIdx is the index of the candidate short-term RPS into the list of candidate short-term RPSs signalled in the active SPS. CurrRpsIdx equal to num_short_term_ref_pic_sets indicates that the short-term RPS of the current picture is directly signalled in the slice header.

For each i in the range of 0 to NumPocLtCurr − 1, inclusive, when CurrDeltaPocMsbPresentFlag[i] is equal to 1, it is a requirement of bitstream conformance that the following conditions apply:

– There shall be no j in the range of 0 to NumPocStCurrBefore − 1, inclusive, for which PocLtCurr[i] is equal to PocStCurrBefore[j].

– There shall be no j in the range of 0 to NumPocStCurrAfter − 1, inclusive, for which PocLtCurr[i] is equal to PocStCurrAfter[j].

– There shall be no j in the range of 0 to NumPocStFoll − 1, inclusive, for which PocLtCurr[i] is equal to PocStFoll[j].

– There shall be no j in the range of 0 to NumPocLtCurr − 1, inclusive, where j is not equal to i, for which PocLtCurr[i] is equal to PocLtCurr[j].

For each i in the range of 0 to NumPocLtFoll − 1, inclusive, when FollDeltaPocMsbPresentFlag[i] is equal to 1, it is a requirement of bitstream conformance that the following conditions apply:

– There shall be no j in the range of 0 to NumPocStCurrBefore − 1, inclusive, for which PocLtFoll[i] is equal to PocStCurrBefore[j].

– There shall be no j in the range of 0 to NumPocStCurrAfter − 1, inclusive, for which PocLtFoll[i] is equal to PocStCurrAfter[j].

– There shall be no j in the range of 0 to NumPocStFoll − 1, inclusive, for which PocLtFoll[i] is equal to PocStFoll[j].

– There shall be no j in the range of 0 to NumPocLtFoll − 1, inclusive, where j is not equal to i, for which PocLtFoll[i] is equal to PocLtFoll[j].

– There shall be no j in the range of 0 to NumPocLtCurr − 1, inclusive, for which PocLtFoll[i] is equal to PocLtCurr[j].

For each i in the range of 0 to NumPocLtCurr − 1, inclusive, when CurrDeltaPocMsbPresentFlag[i] is equal to 0, it is a requirement of bitstream conformance that the following conditions apply:

– There shall be no j in the range of 0 to NumPocStCurrBefore − 1, inclusive, for which PocLtCurr[i] is equal to (PocStCurrBefore[j] & (MaxPicOrderCntLsb − 1)).

– There shall be no j in the range of 0 to NumPocStCurrAfter − 1, inclusive, for which PocLtCurr[i] is equal to (PocStCurrAfter[j] & (MaxPicOrderCntLsb − 1)).

– There shall be no j in the range of 0 to NumPocStFoll − 1, inclusive, for which PocLtCurr[i] is equal to (PocStFoll[j] & (MaxPicOrderCntLsb − 1)).

– There shall be no j in the range of 0 to NumPocLtCurr − 1, inclusive, where j is not equal to i, for which PocLtCurr[i] is equal to (PocLtCurr[j] & (MaxPicOrderCntLsb − 1)).

For each i in the range of 0 to NumPocLtFoll − 1, inclusive, when FollDeltaPocMsbPresentFlag[i] is equal to 0, it is a requirement of bitstream conformance that the following conditions apply:

– There shall be no j in the range of 0 to NumPocStCurrBefore − 1, inclusive, for which PocLtFoll[i] is equal to (PocStCurrBefore[j] & (MaxPicOrderCntLsb − 1)).

– There shall be no j in the range of 0 to NumPocStCurrAfter − 1, inclusive, for which PocLtFoll[i] is equal to (PocStCurrAfter[j] & (MaxPicOrderCntLsb − 1)).

– There shall be no j in the range of 0 to NumPocStFoll − 1, inclusive, for which PocLtFoll[i] is equal to (PocStFoll[j] & (MaxPicOrderCntLsb − 1)).

– There shall be no j in the range of 0 to NumPocLtFoll − 1, inclusive, where j is not equal to i, for which PocLtFoll[i] is equal to (PocLtFoll[j] & (MaxPicOrderCntLsb − 1)).

– There shall be no j in the range of 0 to NumPocLtCurr − 1, inclusive, for which PocLtFoll[i] is equal to (PocLtCurr[j] & (MaxPicOrderCntLsb − 1)).

The variable NumPocTotalCurr is derived as specified in subclause 7.4.7.2. It is a requirement of bitstream conformance that the following applies to the value of NumPocTotalCurr:

– If nuh_layer_id is equal to 0 and the current picture is a BLA picture or a CRA picture, the value of NumPocTotalCurr shall be equal to 0.

– Otherwise, when the current picture contains a P or B slice, the value of NumPocTotalCurr shall not be equal to 0.

The RPS of the current picture consists of five RPS lists; RefPicSetStCurrBefore, RefPicSetStCurrAfter, RefPicSetStFoll, RefPicSetLtCurr and RefPicSetLtFoll. RefPicSetStCurrBefore, RefPicSetStCurrAfter, and RefPicSetStFoll are collectively referred to as the short-term RPS. RefPicSetLtCurr and RefPicSetLtFoll are collectively referred to as the long-term RPS.

NOTE 3 – RefPicSetStCurrBefore, RefPicSetStCurrAfter, and RefPicSetLtCurr contain all reference pictures that may be used for inter prediction of the current picture and one or more pictures that follow the current picture in decoding order. RefPicSetStFoll

and RefPicSetLtFoll consist of all reference pictures that are *not* used for inter prediction of the current picture but may be used in inter prediction for one or more pictures that follow the current picture in decoding order.

The variable currPicLayerId is set to be the nuh_layer_id of the current picture and the derivation process for the RPS and picture marking are performed according to the following ordered steps:

1. The following applies:

```
for( i = 0; i < NumPocLtCurr; i++ )
    if( !CurrDeltaPocMsbPresentFlag[ i ] )
        if( there is a reference picture picX in the DPB with slice_pic_order_cnt_lsb equal to PocLtCurr[ i ]
and nuh_layer_id equal to currPicLayerId)
            RefPicSetLtCurr[ i ] = picX
        else
            RefPicSetLtCurr[ i ] = "no reference picture"
    else
        if( there is a reference picture picX in the DPB with PicOrderCntVal equal to PocLtCurr[ i ] and
nuh_layer_id equal to currPicLayerId)
            RefPicSetLtCurr[ i ] = picX
        else
            RefPicSetLtCurr[ i ] = "no reference picture"                                        (8-6)
for( i = 0; i < NumPocLtFoll; i++ )
    if( !FollDeltaPocMsbPresentFlag[ i ] )
        if( there is a reference picture picX in the DPB with slice_pic_order_cnt_lsb equal to PocLtFoll[ i ] and
nuh_layer_id equal to currPicLayerId)
            RefPicSetLtFoll[ i ] = picX
        else
            RefPicSetLtFoll[ i ] = "no reference picture"
    else
        if( there is a reference picture picX in the DPB with PicOrderCntVal equal to PocLtFoll[ i ] and
nuh_layer_id equal to currPicLayerId)
            RefPicSetLtFoll[ i ] = picX
        else
            RefPicSetLtFoll[ i ] = "no reference picture"
```

2. All reference pictures that are included in RefPicSetLtCurr and RefPicSetLtFoll and with nuh_layer_id equal to currPicLayerId are marked as "used for long-term reference".

3. The following applies:

```
for( i = 0; i < NumPocStCurrBefore; i++ )
    if( there is a short-term reference picture picX in the DPB
            with PicOrderCntVal equal to PocStCurrBefore[ i ] and nuh_layer_id equal to currPicLayerId)
        RefPicSetStCurrBefore[ i ] = picX
    else
        RefPicSetStCurrBefore[ i ] = "no reference picture"

for( i = 0; i < NumPocStCurrAfter; i++ )
    if( there is a short-term reference picture picX in the DPB
            with PicOrderCntVal equal to PocStCurrAfter[ i ] and nuh_layer_id equal to currPicLayerId)
        RefPicSetStCurrAfter[ i ] = picX
    else
        RefPicSetStCurrAfter[ i ] = "no reference picture"                                       (8-7)

for( i = 0; i < NumPocStFoll; i++ )
    if( there is a short-term reference picture picX in the DPB
            with PicOrderCntVal equal to PocStFoll[ i ] and nuh_layer_id equal to currPicLayerId)
        RefPicSetStFoll[ i ] = picX
    else
        RefPicSetStFoll[ i ] = "no reference picture"
```

4. All reference pictures in the DPB that are not included in RefPicSetLtCurr, RefPicSetLtFoll, RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetStFoll and with nuh_layer_id equal to currPicLayerId are marked as "unused for reference".

NOTE 4 – There may be one or more entries in the RPS lists that are equal to "no reference picture" because the corresponding pictures are not present in the DPB. Entries in RefPicSetStFoll or RefPicSetLtFoll that are equal to "no reference picture" should

be ignored. An unintentional picture loss should be inferred for each entry in RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetLtCurr that is equal to "no reference picture".

NOTE 5 – A picture cannot be included in more than one of the five RPS lists.

It is a requirement of bitstream conformance that the RPS is restricted as follows:

- There shall be no entry in RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetLtCurr for which one or more of the following are true:
  - The entry is equal to "no reference picture".
  - The entry is a sub-layer non-reference picture and has TemporalId equal to that of the current picture.
  - The entry is a picture that has TemporalId greater than that of the current picture.
- There shall be no entry in RefPicSetLtCurr or RefPicSetLtFoll for which the difference between the picture order count value of the current picture and the picture order count value of the entry is greater than or equal to 2^{24}.
- When the current picture is a TSA picture, there shall be no picture included in the RPS with TemporalId greater than or equal to the TemporalId of the current picture.
- When the current picture is an STSA picture, there shall be no picture included in RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetLtCurr that has TemporalId equal to that of the current picture.
- When the current picture is a picture that follows, in decoding order, an STSA picture that has TemporalId equal to that of the current picture, there shall be no picture that has TemporalId equal to that of the current picture included in RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetLtCurr that precedes the STSA picture in decoding order.
- When the current picture is a CRA picture, there shall be no picture included in the RPS that precedes, in decoding order, any preceding IRAP picture in decoding order (when present).
- When the current picture is a trailing picture, there shall be no picture in RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetLtCurr that was generated by the decoding process for generating unavailable reference pictures as specified in subclause 8.3.3.
- When the current picture is a trailing picture, there shall be no picture in the RPS that precedes the associated IRAP picture in output order or decoding order.
- When the current picture is a RADL picture, there shall be no picture included in RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetLtCurr that is any of the following:
  - A RASL picture
  - A picture that was generated by the decoding process for generating unavailable reference pictures as specified in subclause 8.3.3
  - A picture that precedes the associated IRAP picture in decoding order
- When the sps_temporal_id_nesting_flag is equal to 1, the following applies:
  - Let tIdA be the value of TemporalId of the current picture picA.
  - Any picture picB with TemporalId equal to tIdB that is less than or equal to tIdA shall not be included in RefPicSetStCurrBefore, RefPicSetStCurrAfter, or RefPicSetLtCurr of picA when there exists a picture picC that has TemporalId less than tIdB, follows picB in decoding order, and precedes picA in decoding order.

Annex D

Supplemental enhancement information

(This annex forms an integral part of this Recommendation | International Standard)

D.1 SEI payload syntax

D.1.1 General SEI message syntax

sei_payload(payloadType, payloadSize) {	**Descriptor**
if(nal_unit_type = = PREFIX_SEI_NUT)	
if(payloadType = = 0)	
...	
else if(payloadType = = XXX)	
layer_dependency_change(payloadSize)	
else if(payloadType = = XXX)	
layers_present(payloadSize)	
else if(payloadType = = XXX)	
three_dimensional_reference_displays_info(payloadSize)	
...	
else	
reserved_sei_message(payloadSize)	
else /* nal_unit_type = = SUFFIX_SEI_NUT */	
if(payloadType = = 3)	
filler_payload(payloadSize)	
...	
else	
reserved_sei_message(payloadSize)	
if(more_data_in_payload( )) {	
if(payload_extension_present( ))	
reserved_payload_extension_data	u(v)
payload_bit_equal_to_one /* equal to 1 */	f(1)
while(!byte_aligned( ))	
payload_bit_equal_to_zero /* equal to 0 */	f(1)
}	
}	

[Ed. (GT) Added SEI as prefix SEI. It needs to be clarified if this is accoriding to adoptions. Moreover values of payloadType need to be clarified.]

Annex F

Common syntax, semantics and decoding processes for multi-layer video coding extensions

(This annex forms an integral part of this Recommendation | International Standard)

This annex specifies the common syntax, semantics and decoding processes for multi-layer video coding extensions.

F.1 Scope

Common syntax, semantics and decoding processes for multi-layer video coding extensions are specified in this annex with reference made to clauses 2-9 and Annexes A-E and G.

F.2 Normative references

The specifications in clause 2 apply.

F.3 Definitions

For the purpose of this annex, the following definitions apply in addition to the definitions in clause 3. These definitions are either not present in clause 3 or replace definitions in clause 3.

[Ed. (YK&MH&CY): Definitions should be checked and potentially refined, including: BLA AU, IDR AU, CRA AU, coded video sequence, output order, RADL AU, RASL AU, IRAP AU, IRAP picture, (reference picture), STSA AU, TSA AU]

F.3.1 **access unit:** A set of *NAL units* that are associated with each other according to a specified classification rule, are consecutive in *decoding order*, and contain the *VCL NAL units* of all *coded pictures* associated with the same output time and their associated non-VCL NAL units. [Ed. (MH): could the use of "output time" be avoided in the definition. Why not to use picture order count instead?]

NOTE – Pictures in the same access unit are associated with the same picture order count.

F.3.2 **associated IRAP picture:** The previous *IRAP picture* in *decoding order* within the same *layer* (if present).

F.3.3 **base layer:** A *layer* in which all *VCL NAL units* have nuh_layer_id equal to 0.

F.3.4 **broken link access (BLA) access unit:** An *access unit* in which all the *coded pictures* are *BLA pictures*.

F.3.5 **coded picture:** A *coded representation* of a *picture* comprising *VCL NAL units* with a particular value of nuh_layer_id within an *access unit* and containing all *coding tree units* of the *picture*. [Ed. (CY): consider defining picture by associating nuh_layer_id. In HEVC base, picture is defined as arrays of luma and chroma samples, however, it is often associated with other properties, e.g., coding tree units. So to be absolutely precise, it might be clearer and applicable to define picture as follows: *picture*: An array of *luma* samples in monochrome format or an array of *luma* samples and two corresponding arrays of *chroma* samples in 4:2:0, 4:2:2, and 4:4:4 colour format with the same value of nuh_layer_id.]

F.3.6 **collocated sample:** A sample TBD. [Ed. (GT) Maybe it is easier to define a collocated position and require collocated samples to have it?]

F.3.7 **direct reference layer:** A *layer* which may be used for inter-layer prediction of another *layer*.

F.3.8 **inter-layer prediction:** A *prediction* in manner that is dependent on data elements (e.g. sample values or motion vectors) of *reference pictures* with another value of nuh_layer_id than that for the current *picture.*

F.3.9 **instantaneous decoding refresh (IDR) access unit:** An *access unit* in which all the *coded pictures* are *IDR pictures*.

F.3.10 **leading picture:** A *picture* that is in the same *layer* as the *associated IRAP picture* and precedes the *associated IRAP picture* in *output order*.

F.3.11 **non-base layer:** A *layer* in which all *VCL NAL units* have the same nuh_layer_id value greater than 0.

F.3.12 **picture order count:** A variable that is associated with each *picture*, uniquely identifies the associated *picture* with a particular value of nuh_layer_id among all *pictures* with that same particular value of nuh_layer_id in the *CVS*, and, when the associated *picture* is to be output from the *decoded picture buffer*, indicates the position of the associated *picture* in *output order* relative to the *output order* positions of the other *pictures*

with that same particular value of nuh_layer_id in the same *CVS* that are to be output from the *decoded picture buffer*.

F.3.13 **reference layer picture:** A *picture* in a *direct reference layer* which is used for inter-layer prediction of the current *picture* and is in the same access unit as the *current picture*.

F.3.14 **reference picture list:** A list of reference pictures that is used for inter prediction or inter-layer prediction of a P or B slice.

F.3.15 **target output layer:** A *layer* that is to be output.

F.3.16 **trailing picture:** A *picture* that is in the same *layer* as the *associated IRAP picture* and follows the *associated IRAP picture* in *output order*.

F.3.17 **output time:** A time instance when a *decoded picture* is to be output and derived by the specifications in Annex C, if the timing information is present in the *coded video sequence*. [Ed. (CY): Consider clearer definition of "output time"] [Ed. (MH): Term "output time" is used in many occasions in HEVC v1, but it is not included in the definitions in clause 3. If its definition is considered necessary, the definition should IMO appear in clause 3 and be applicable to HEVC v1 too.]

F.3.18 **view:** A sequence of pictures with an identical value of ViewId.

NOTE – A view typically represents a sequence of pictures captured with one camera.

F.4 Abbreviations

The specification in clause 4 apply.

F.5 Conventions

The specification in clause 5 apply.

F.6 Source, coded, decoded and output data formats, scanning processes, and neighbouring relationships

The specification in clause 6 apply.

F.7 Syntax and semantics

This clause specifies syntax and semantics for CVSs that conform to one or more of the profiles specified in this annex.

F.7.1 Method of specifying syntax in tabular form

The specifications in subclause 7.1 apply.

F.7.2 Specification of syntax functions, categories, and descriptors

The specifications in subclause 7.2 apply.

F.7.3 Syntax in tabular form

F.7.3.1 NAL unit syntax

The specifications in subclause 7.3.1 apply.

F.7.3.1.1 General NAL unit syntax

The specifications in subclause 7.3.1.1 apply.

F.7.3.1.2 NAL unit header syntax

The specifications in subclause 7.3.1.2 apply.

F.7.3.2 Raw byte sequence payloads and RBSP trailing bits syntax

F.7.3.2.1 Video parameter set RBSP

video_parameter_set_rbsp( ) {	**Descriptor**
vps_video_parameter_set_id	u(4)
vps_reserved_three_2bits	u(2)
vps_max_layers_minus1	u(6)
vps_max_sub_layers_minus1	u(3)
vps_temporal_id_nesting_flag	u(1)
vps_extension_offset //vps_reserved_0xffff_16bits	u(16)
profile_tier_level(1, vps_max_sub_layers_minus1)	
vps_sub_layer_ordering_info_present_flag	u(1)
for(i = (vps_sub_layer_ordering_info_present_flag ? 0 : vps_max_sub_layers_minus1); i <= vps_max_sub_layers_minus1; i++) {	
vps_max_dec_pic_buffering_minus1[i]	ue(v)
vps_max_num_reorder_pics[i]	ue(v)
vps_max_latency_increase_plus1[i]	ue(v)
}	
vps_max_layer_id	u(6)
vps_num_layer_sets_minus1	ue(v)
for(i = 1; i <= vps_num_layer_sets_minus1; i++)	
for(j = 0; j <= vps_max_layer_id; j++)	
layer_id_included_flag[i][j]	u(1)
vps_timing_info_present_flag	u(1)
if(vps_timing_info_present_flag) {	
vps_num_units_in_tick	u(32)
vps_time_scale	u(32)
vps_poc_proportional_to_timing_flag	u(1)
if(vps_poc_proportional_to_timing_flag)	
vps_num_ticks_poc_diff_one_minus1	ue(v)
vps_num_hrd_parameters	ue(v)
for(i = 0; i < vps_num_hrd_parameters; i++) {	
hrd_layer_set_idx[i]	ue(v)
if(i > 0)	
cprms_present_flag[i]	u(1)
hrd_parameters(cprms_present_flag[i], vps_max_sub_layers_minus1)	
}	
}	
vps_extension_flag	u(1)
if(vps_extension_flag) {	
vps_extension( )	
vps_extension2_flag	u(1)
if(vps_extension2_flag)	
while(more_rbsp_data( ))	
vps_extension_data_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

F.7.3.2.1.1 Video parameter set extension syntax

vps_extension( ) {	Descriptor
while(!byte_aligned( ))	
vps_extension_byte_alignment_reserved_one_bit	u(1)
avc_base_layer_flag	u(1)
splitting_flag	u(1)
for(i = 0, NumScalabilityTypes = 0; i < 16; i++) {	
scalability_mask[i]	u(1)
NumScalabilityTypes += scalability_mask[i]	
}	
for(j = 0; j < (NumScalabilityTypes − splitting_flag); j++)	
dimension_id_len_minus1[j]	u(3)
vps_nuh_layer_id_present_flag	u(1)
for(i = 0; i <= vps_max_layers_minus1; i++) {	
if(vps_nuh_layer_id_present_flag && i > 0) [Ed. (JB): syntax is not compatible with SHVC, or use of splitting_flag.]	
layer_id_in_nuh[i]	u(6)
if(!splitting_flag)	
for(j = 0; j < NumScalabilityTypes; j++)	
dimension_id[i][j]	u(v)
}	
for(i = 1; i <= vps_max_layers_minus1; i++)	
for(j = 0; j < i; j++)	
direct_dependency_flag[i][j]	u(1)
for(i = 0; i< vps_max_layers_minus1; i++)	
max_tid_il_ref_pics_plus1[i]	u(3)
vps_number_layer_sets_minus1	u(10)
vps_num_profile_tier_level_minus1	u(6)
for(i = 1; i <= vps_num_profile_tier_level_minus1; i ++) {	
vps_profile_present_flag[i]	u(1)
if(!vps_profile_present_flag[i])	
profile_ref_minus1[i]	u(6)
profile_tier_level(vps_profile_present_flag[i], vps_max_sub_layers_minus1)	
}	
numOutputLayerSets = vps_number_layer_sets_minus1 + 1	
more_output_layer_sets_than_default_flag	u(1)
if(more_output_layer_sets_than_default_flag) {	
num_add_output_layer_sets_minus1	u(10)
numOutputLayerSets += num_add_output_layer_sets_minus1 + 1	
}	
if(numOutputLayerSets > 1)	
default_one_target_output_layer_flag	u(1)
for(i = 1; i < numOutputLayerSets; i++) {	
if(i > vps_number_layer_sets_minus1) {	
output_layer_set_idx_minus1[i]	u(v)
lsIdx = output_layer_set_idx_minus1[i] + 1	

for(j = 0 ; j < NumLayersInIdList[lsIdx] − 1; j++)	
output_layer_flag[i][j]	u(1)
}	
profile_level_tier_idx[i]	u(v)
}	
max_one_active_ref_layer_flag	u(1)
direct_dep_type_len_minus2	ue(v)
for(i = 1; i <= vps_max_layers_minus1; i++)	
for(j = 0; j < i; j++)	
if(direct_dependency_flag[i][j])	
direct_dependency_type[i][j]	u(v)
vps_shvc_reserved_zero_flag	u(1)
}	

[Ed. (YK): Align the syntax table style with other syntax tables, e.g. each row should "keep with next".]

F.7.3.2.2 Sequence parameter set RBSP syntax

seq_parameter_set_rbsp( ) {	**Descriptor**
sps_video_parameter_set_id	u(4)
if(nuh_layer_id == 0) {	
sps_max_sub_layers_minus1	u(3)
sps_temporal_id_nesting_flag	u(1)
profile_tier_level(1, sps_max_sub_layers_minus1)	
}	
sps_seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
conformance_window_flag	u(1)
if(conformance_window_flag) {	
conf_win_left_offset	ue(v)
conf_win_right_offset	ue(v)
conf_win_top_offset	ue(v)
conf_win_bottom_offset	ue(v)
}	
bit_depth_luma_minus8	ue(v)
bit_depth_chroma_minus8	ue(v)
log2_max_pic_order_cnt_lsb_minus4	ue(v)
sps_sub_layer_ordering_info_present_flag	u(1)
for(i = (sps_sub_layer_ordering_info_present_flag ? 0 : sps_max_sub_layers_minus1); i <= sps_max_sub_layers_minus1; i++) {	
sps_max_dec_pic_buffering_minus1[i]	ue(v)
sps_max_num_reorder_pics[i]	ue(v)
sps_max_latency_increase_plus1[i]	ue(v)
}	
log2_min_luma_coding_block_size_minus3	ue(v)
log2_diff_max_min_luma_coding_block_size	ue(v)
log2_min_transform_block_size_minus2	ue(v)
log2_diff_max_min_transform_block_size	ue(v)
max_transform_hierarchy_depth_inter	ue(v)
max_transform_hierarchy_depth_intra	ue(v)
scaling_list_enabled_flag	u(1)
if(scaling_list_enabled_flag) {	
sps_scaling_list_data_present_flag	u(1)
if(sps_scaling_list_data_present_flag)	
scaling_list_data( )	
}	
amp_enabled_flag	u(1)
sample_adaptive_offset_enabled_flag	u(1)
pcm_enabled_flag	u(1)
if(pcm_enabled_flag) {	
pcm_sample_bit_depth_luma_minus1	u(4)

pcm_sample_bit_depth_chroma_minus1	u(4)
log2_min_pcm_luma_coding_block_size_minus3	ue(v)
log2_diff_max_min_pcm_luma_coding_block_size	ue(v)
pcm_loop_filter_disabled_flag	u(1)
}	
num_short_term_ref_pic_sets	ue(v)
for(i = 0; i < num_short_term_ref_pic_sets; i++)	
short_term_ref_pic_set(i)	
long_term_ref_pics_present_flag	u(1)
if(long_term_ref_pics_present_flag) {	
num_long_term_ref_pics_sps	ue(v)
for(i = 0; i < num_long_term_ref_pics_sps; i++) {	
lt_ref_pic_poc_lsb_sps[i]	u(v)
used_by_curr_pic_lt_sps_flag[i]	u(1)
}	
}	
sps_temporal_mvp_enabled_flag	u(1)
strong_intra_smoothing_enabled_flag	u(1)
vui_parameters_present_flag	u(1)
if(vui_parameters_present_flag)	
vui_parameters( )	
sps_extension_flag	u(1)
if(sps_extension_flag) {	
sps_extension( )	
sps_extension2_flag	u(1)
if(sps_extension2_flag)	
while(more_rbsp_data( ))	
sps_extension_data_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

F.7.3.2.2.1 Sequence parameter set extension syntax

sps_extension( ) {	**Descriptor**
inter_view_mv_vert_constraint_flag	u(1)
sps_extension_vui_parameters( )	
sps_shvc_reserved_zero_idc	ue(v)
}	

F.7.3.2.3 Picture parameter set RBSP syntax

The specifications in subclause 7.3.2.3 apply.

F.7.3.2.4 Supplemental enhancement information RBSP syntax

The specifications in subclause 7.3.2.4 apply.

F.7.3.2.5 Access unit delimiter RBSP syntax

The specifications in subclause 7.3.2.5 apply.

F.7.3.2.6 End of sequence RBSP syntax

The specifications in subclause 7.3.2.6 apply.

F.7.3.2.7 End of bitstream RBSP syntax

The specifications in subclause 7.3.2.7 apply.

F.7.3.2.8 Filler data RBSP syntax

The specifications in subclause 7.3.2.8 apply.

F.7.3.2.9 Slice segment layer RBSP syntax

The specifications in subclause 7.3.2.9 apply.

F.7.3.2.10 RBSP slice segment trailing bits syntax

The specifications in subclause 7.3.2.10 apply.

F.7.3.2.11 RBSP trailing bits syntax

The specifications in subclause 7.3.2.11 apply.

F.7.3.2.12 Byte alignment syntax

The specifications in subclause 7.3.2.12 apply.

F.7.3.3 Profile, tier and level syntax

profile_tier_level(profilePresentFlag, maxNumSubLayersMinus1) {	**Descriptor**
if(profilePresentFlag) {	
general_profile_space	u(2)
general_tier_flag	u(1)
general_profile_idc	u(5)
for(j = 0; j < 32; j++)	
general_profile_compatibility_flag[j]	u(1)
general_progressive_source_flag	u(1)
general_interlaced_source_flag	u(1)
general_non_packed_constraint_flag	u(1)
general_frame_only_constraint_flag	u(1)
general_reserved_zero_44bits	u(44)
}	
general_level_idc	u(8)
for(i = 0; i < maxNumSubLayersMinus1; i++) {	
sub_layer_profile_present_flag[i]	u(1)
sub_layer_level_present_flag[i]	u(1)
}	
if(maxNumSubLayersMinus1 > 0)	
for(i = maxNumSubLayersMinus1; i < 8; i++)	
reserved_zero_2bits[i]	u(2)
for(i = 0; i < maxNumSubLayersMinus1; i++) {	
if(sub_layer_profile_present_flag[i]) {	
sub_layer_profile_space[i]	u(2)
sub_layer_tier_flag[i]	u(1)
sub_layer_profile_idc[i]	u(5)
for(j = 0; j < 32; j++)	
sub_layer_profile_compatibility_flag[i][j]	u(1)
sub_layer_progressive_source_flag[i]	u(1)
sub_layer_interlaced_source_flag[i]	u(1)
sub_layer_non_packed_constraint_flag[i]	u(1)
sub_layer_frame_only_constraint_flag[i]	u(1)
sub_layer_reserved_zero_44bits[i]	u(44)
}	
if(sub_layer_level_present_flag[i])	
sub_layer_level_idc[i]	u(8)
}	
}	

F.7.3.4 Scaling list data syntax

The specifications in subclause 7.3.4 apply

F.7.3.5 Supplemental enhancement information message syntax

The specifications in subclause 7.3.5 apply

F.7.3.6 Slice segment header syntax

F.7.3.6.1 General slice segment header syntax

slice_segment_header( ) {	**Descriptor**
first_slice_segment_in_pic_flag	u(1)
if(nal_unit_type >= BLA_W_LP && nal_unit_type <= RSV_IRAP_VCL23)	
no_output_of_prior_pics_flag	u(1)
slice_pic_parameter_set_id	ue(v)
if(!first_slice_segment_in_pic_flag) {	
if(dependent_slice_segments_enabled_flag)	
dependent_slice_segment_flag	u(1)
slice_segment_address	u(v)
}	
if(!dependent_slice_segment_flag) {	
if(num_extra_slice_header_bits > 0)	
discardable_flag	u(1)
for(i = 1; i < num_extra_slice_header_bits ; i++)	
slice_reserved_flag[i]	u(1)
slice_type	ue(v)
if(output_flag_present_flag)	
pic_output_flag	u(1)
if(separate_colour_plane_flag = = 1)	
colour_plane_id	u(2)
if(nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) {	
slice_pic_order_cnt_lsb	u(v)
short_term_ref_pic_set_sps_flag	u(1)
if(!short_term_ref_pic_set_sps_flag)	
short_term_ref_pic_set(num_short_term_ref_pic_sets)	
else if(num_short_term_ref_pic_sets > 1)	
short_term_ref_pic_set_idx	u(v)
if(long_term_ref_pics_present_flag) {	
if(num_long_term_ref_pics_sps > 0)	
num_long_term_sps	ue(v)
num_long_term_pics	ue(v)
for(i = 0; i < num_long_term_sps + num_long_term_pics; i++) {	
if(i < num_long_term_sps) {	
if(num_long_term_ref_pics_sps > 1)	
lt_idx_sps[i]	u(v)
} else {	
poc_lsb_lt[i]	u(v)
used_by_curr_pic_lt_flag[i]	u(1)
}	
delta_poc_msb_present_flag[i]	u(1)
if(delta_poc_msb_present_flag[i])	
delta_poc_msb_cycle_lt[i]	ue(v)
}	
}	
if(sps_temporal_mvp_enabled_flag)	
slice_temporal_mvp_enabled_flag	u(1)
}	
if(nuh_layer_id > 0 && NumDirectRefLayers[nuh_layer_id] > 0) {	

inter_layer_pred_enabled_flag	u(1)
if(inter_layer_pred_enabled_flag && NumDirectRefLayers[nuh_layer_id] > 1) {	
if(!max_one_active_ref_layer_flag)	
num_inter_layer_ref_pics_minus1	u(v)
for(i = 0; i < NumActiveRefLayerPics; i++)	
inter_layer_pred_layer_idc[i]	u(v)
}	
}	
if(NumSamplePredRefLayers[nuh_layer_id] > 0 && NumActiveRefLayerPics > 0)	
inter_layer_sample_pred_only_flag	u(1)
if(sample_adaptive_offset_enabled_flag) {	
slice_sao_luma_flag	u(1)
slice_sao_chroma_flag	u(1)
}	
if(slice_type = = P || slice_type = = B) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag) {	
num_ref_idx_l0_active_minus1	ue(v)
if(slice_type = = B)	
num_ref_idx_l1_active_minus1	ue(v)
}	
if(lists_modification_present_flag && NumPocTotalCurr > 1)	
ref_pic_lists_modification( )	
if(slice_type = = B)	
mvd_l1_zero_flag	u(1)
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(slice_temporal_mvp_enabled_flag) {	
if(nuh_layer_id > 0 && NumActiveMotionPredRefLayers > 0)	
alt_collocated_indication_flag	u(1)
if(alt_collocated_indication_flag && NumActiveMotionPredRefLayers > 1)	
collocated_ref_layer_idx	ue(v)
else {	
if(slice_type = = B)	
collocated_from_l0_flag	u(1)
if((collocated_from_l0_flag && num_ref_idx_l0_active_minus1 > 0) || (!collocated_from_l0_flag && num_ref_idx_l1_active_minus1 > 0))	
collocated_ref_idx	ue(v)
}	
}	
if((weighted_pred_flag && slice_type = = P) || (weighted_bipred_flag && slice_type = = B))	
pred_weight_table( )	
five_minus_max_num_merge_cand	ue(v)
}	
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)

}	
if(deblocking_filter_override_enabled_flag)	
deblocking_filter_override_flag	u(1)
if(deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(pps_loop_filter_across_slices_enabled_flag && (slice_sao_luma_flag || slice_sao_chroma_flag || !slice_deblocking_filter_disabled_flag))	
slice_loop_filter_across_slices_enabled_flag	u(1)
}	
if(tiles_enabled_flag || entropy_coding_sync_enabled_flag) {	
num_entry_point_offsets	ue(v)
if(num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset_minus1[i]	u(v)
}	
}	
if(slice_segment_header_extension_present_flag) {	
slice_segment_header_extension_length	ue(v)
for(i = 0; i < slice_segment_header_extension_length; i++)	
slice_segment_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

F.7.3.6.2 Reference picture list modification syntax

The specifications in subclause 7.3.6.2 apply

F.7.3.6.3 Weighted prediction parameters syntax

The specifications in subclause 7.3.6.3 apply

F.7.3.7 Short-term reference picture set syntax

The specifications in subclause 7.3.7 apply

F.7.3.8 Slice segment data syntax

F.7.3.8.1 General slice segment data syntax

The specifications in subclause 7.3.8.1 apply.

F.7.3.8.2 Coding tree unit syntax

The specifications in subclause 7.3.8.2 apply.

F.7.3.8.3 Sample adaptive offset syntax

The specifications in subclause 7.3.8.3 apply.

F.7.3.8.4 Coding quadtree syntax

The specifications in subclause 7.3.8.4 apply.

F.7.3.8.5 Coding unit syntax

The specifications in subclause 7.3.8.5 apply.

F.7.3.8.6 Prediction unit syntax

The specifications in subclause 7.3.8.6 apply.

F.7.3.8.7 PCM sample syntax

The specifications in subclause 7.3.8.7 apply.

F.7.3.8.8 Transform tree syntax

The specifications in subclause 7.3.8.8 apply.

F.7.3.8.9 Motion vector difference syntax

The specifications in subclause 7.3.8.9 apply.

F.7.3.8.10 Transform unit syntax

The specifications in subclause 7.3.8.10 apply.

F.7.3.8.11 Residual coding syntax

The specifications in subclause 7.3.8.11 apply.

F.7.4 Semantics

F.7.4.1 General

F.7.4.2 NAL unit semantics

F.7.4.2.1 General NAL unit semantics

The specifications in subclause 7.4.2.1 apply.

F.7.4.2.2 NAL unit header semantics

The specifications in subclause 7.4.2.2 apply with following modifications and additions.

nuh_layer_id specifies the identifier of the layer.

When the nal_unit_type value nalUnitTypeA is equal to IDR_W_DLP, IDR_N_LP, BLA_W_LP, BLA_W_DLP or BLA_N_LP for a coded picture, the nal_unit_type value shall be equal to nalUnitTypeA for all VCL NAL units of all coded pictures of the same access unit.

When nal_unit_type is equal to AUD_NUT, the value of nuh_layer_id shall be equal to the minimum of the nuh_layer_id values of all VCL NAL units in the access unit.

F.7.4.2.3 Encapsulation of an SODB within an RBSP (informative)

The specifications in subclause 7.4.2.3 apply.

F.7.4.2.4 Order of NAL units and association to coded pictures, access units, and coded video sequences

F.7.4.2.4.1 General

The specifications in subclause 7.4.2.4.1 apply with the following additions.

A coded picture with nuh_layer_id equal to nuhLayerIdA shall precede in decoding order all coded pictures with nuh_layer_id greater than nuhLayerIdA in the same access unit.

F.7.4.2.4.2 Order of VPS, SPS and PPS RBSPs and their activation

The specifications in subclause 7.4.2.4.2 apply with the following additions.

Each PPS RBSP is initially considered not active for any layer with nuh_layer_id greater than 0 at the start of the operation of the decoding process. At most one PPS RBSP is considered active for each non-zero nuh_layer_id value at

any given moment during the operation of the decoding process, and the activation of any particular PPS RBSP for a particular non-zero nuh_layer_id value results in the deactivation of the previously-active PPS RBSP for that non-zero nuh_layer_id value (if any).

When a PPS RBSP (with a particular value of pps_pic_parameter_set_id) is not active for a nuh_layer_id value and it is referred to by a coded slice segment NAL unit (using a value of slice_pic_parameter_set_id equal to the pps_pic_parameter_set_id and having that value of nuh_layer_id), it is activated for that nuh_layer_id value. This PPS RBSP is called the active layer PPS RBSP for that nuh_layer_id value until it is deactivated by the activation of another PPS RBSP for the same layer. A PPS RBSP, with that particular value of pps_pic_parameter_set_id, shall be available to the decoding process prior to its activation, included in at least one access unit with TemporalId less than or equal to the TemporalId of the PPS NAL unit or provided through external means. The nuh_layer_id value of the NAL unit containing the PPS RBSP that is activated for nuh_layer_id equal to nuhLayerIdA shall be less than or equal to nuhLayerIdA. The same PPS RBSP may be the active layer PPS for more than one nuh_layer_id value.

Any PPS NAL unit containing the value of pps_pic_parameter_set_id for the active layer PPS RBSP for a coded picture shall have the same content as that of the active layer PPS RBSP for the coded picture, unless it follows the last VCL NAL unit of the coded picture and precedes the first VCL NAL unit of another coded picture.

Each SPS RBSP is initially considered not active for any layer with nuh_layer_id greater than 0 at the start of the operation of the decoding process. At most one SPS RBSP is considered active for each non-zero nuh_layer_id value at any given moment during the operation of the decoding process, and the activation of any particular SPS RBSP for a particular non-zero nuh_layer_id value results in the deactivation of the previously-active SPS RBSP for that non-zero nuh_layer_id value (if any).

When an SPS RBSP (with a particular value of sps_seq_parameter_set_id) is not already active for a nuh_layer_id value and it is referred to by activation of a PPS RBSP for that nuh_layer_id value (in which pps_seq_parameter_set_id is equal to the sps_seq_parameter_set_id value), it is activated for that nuh_layer_id value. This SPS RBSP is called the active layer SPS RBSP for that nuh_layer_id value until it is deactivated by the activation of another SPS RBSP for the same layer. An SPS RBSP, with that particular value of sps_seq_parameter_set_id shall be available to the decoding process prior to its activation, included in at least one access unit with TemporalId equal to 0 or provided through external means. An activated SPS RBSP for a particular nuh_layer_id value shall remain active for a sequence of pictures in decoding order with that nuh_layer_id value starting from an IRAP picture with NoRaslOutputFlag equal to 1 having that nuh_layer_id value, inclusive, until either the next IRAP picture with that nuh_layer_id value and NoRaslOutputFlag equal to 1, exclusive, or the end of the CVS, whichever is earlier. The nuh_layer_id value the NAL unit containing the SPS RBSP that is activated for nuh_layer_id equal to nuhLayerIdA shall be less than or equal to nuhLayerIdA. The same SPS RBSP may be the active layer SPS for more than one nuh_layer_id value.

Any SPS NAL unit containing the value of sps_seq_parameter_set_id for the active layer SPS RBSP shall have the same content as that of the active layer SPS RBSP unless it follows the last coded picture for which the active layer SPS is required to be active and precedes the first NAL unit activating a SPS of the same value of seq_parameter_set_id.

During operation of the decoding process for VCL NAL units with a non-zero nuh_layer_id value, the values of parameters of the active layer SPS for that non-zero nuh_layer_id value, and the active layer PPS RBSP for that non-zero nuh_layer_id value are considered in effect.

F.7.4.2.4.3 Order of access units and their association to CVS

The specifications in subclause 7.4.2.4.3 apply.

F.7.4.2.4.4 Order of NAL units and coded pictures and association to access units

The specifications in subclause 7.4.2.4.4 apply.

F.7.4.2.4.5 Order of VCL NAL units and association to coded pictures

The specifications in subclause 7.4.2.4.5 apply.

F.7.4.3 Raw byte sequence payloads, trailing bits, and byte alignment semantics

F.7.4.3.1 Video parameter set RBSP semantics

The specifications in subclause 7.4.3.1 apply with following modifications and additions.

– layerSetLayerIdList is replaced by LayerSetLayerIdList

– numLayersInIdList is replaced by NumLayersInIdList

vps_extension_offset specifies the byte offset of the next set of fixed-length coded information in the VPS NAL unit, starting from the beginning of the NAL unit.

NOTE –VPS information for non-base layer or view starts from a byte-aligned position of the VPS NAL unit, with fixed-length coded information that is essential for session negotiation and/or capability exchange. The byte offset specified by vps_extension_offset would then help to locate and access those essential information in the VPS NAL unit without the need of entropy decoding, which may not be equipped with some network entities that may desire to access only the information in the VPS that is essential for session negotiation and/or capability exchange.

vps_extension_flag equal to 0 specifies that no vps_extension( ) syntax structure is present in the VPS RBSP syntax structure. vps_extension_flag equal to 1 specifies that the vps_extension( ) syntax structure is present in the VPS RBSP syntax structure. When vps_max_layers_minus1 is greater than 0, vps_extension_flag shall be equal to 1.

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 shall be equal to 0 in bitstreams conforming to this version of this Specification. The value of 1 for vps_extension2_flag is reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore all data that follow the value 1 for vps_extension2_flag in a VPS NAL unit.

F.7.4.3.1.1 Video parameter set extension semantics

vps_extension_byte_alignment_reserved_one_bit shall be equal to 1.

avc_base_layer_flag equal to 1 specifies that the base layer conforms to Rec. ITU-T H.264 | ISO/IEC 14496-10, equal to 0 specifies that it conforms to this specification.

[Ed. (YK): For possible support of base layer of other codecs, e.g. MPEG-2, a flag is not sufficient.]

When avc_base_layer_flag equal to 1, in the Rec. ITU-T H.264 | ISO/IEC 14496-10 conforming base layer, after applying the Rec. ITU-T H.264 | ISO/IEC 14496-10 decoding process for reference picture lists construction the output reference picture lists refPicList0 and refPicList1 (when applicable) does not contain any pictures for which the TemporalId is greater than TemporalId of the coded picture. All sub-bitstreams of the Rec. ITU-T H.264 | ISO/IEC 14496-10 conforming base layer, that can be derived using the sub-bitstream extraction process as specified in Rec. ITU-T H.264 | ISO/IEC 14496-10 subclause G.8.8.1 with any value for temporal_id as the input shall result in a set of CVSs, with each CVS conforming to one or more of the profiles specified in Rec. ITUT H.264 | ISO/IEC 14496-10 Annexes A, G and H.

splitting_flag equal to 1 indicates that the bits of the nuh_layer_id syntax element in the NAL unit header are split into n segments with a length, in bits, according to the values of the dimension_id_len_minus1[i] syntax element and that the n segments are associated with the n scalability dimensions indicated in scalability_mask_flag[i]. When splitting_flag is equal to 1, the value of the j-th segment of the nuh_layer_id of i-th layer shall be equal to the value of dimension_id[i][j]. splitting_flag equal to 0 does not indicate the above constraint.

NOTE 1 – When splitting_flag is equal to 1, i.e. the restriction reported in the semantics of the dimension_id[i][j] syntax element is obeyed, scalable identifiers can be derived from the nuh_layer_id syntax element in the NAL unit header by a bit masked copy as an alternative to the derivation as reported in the semantics of the dimension_id[i][j] syntax element. The respective bit mask for the i-th scalable dimension is defined by the value of the dimension_id_len_minus1[i] syntax element and dimBitOffset[i] as specified in the semantics of dimension_id_len_minus1[j].

scalability_mask[i] equal to 1 indicates that dimension_id syntax elements corresponding to the i-th scalability dimension in Table F-1 are present. scalability_mask[i] equal to 0 indicates that dimension_id syntax elements corresponding to the i-th scalability dimension are not present.

Table F-1 – Mapping of ScalabiltyId to scalability dimensions

scalability_mask index	Scalability dimension	ScalabilityId mapping
0	multiview	ViewId
1-15	Reserved	

dimension_id_len_minus1[j] plus 1 specifies the length, in bits, of the dimension_id[i][j] syntax element.

The variable dimBitOffset[0] is set equal to 0 and for j in the range of 1 to (NumScalabilityTypes − splitting_flag), inclusive, dimBitOffset[j] is derived as follows.

$$\mathrm{dimBitOffset}[\,j\,] = \sum_{\mathrm{dimIdx}=0}^{j-1} (\mathrm{dimension\_id\_len\_minus1}[\,\mathrm{dimIdx}\,] + 1) \qquad \text{(F-1)}$$

When dimension_id_len_minus1[NumScalabilityTypes − 1] is not present, the following applies:

– The value of dimension_id_len_minus1[NumScalabilityTypes − 1] is inferred to be equal to 5 − dimBitOffset[NumScalabilityTypes − 1].

– The value of dimBitOffset[NumScalabilityTypes] is set equal to 6.

vps_nuh_layer_id_present_flag specifies whether the layer_id_in_nuh[i] syntax is present.

layer_id_in_nuh[i] specifies the value of the nuh_layer_id syntax element in VCL NAL units of the i-th layer. For i in a range from 0 to vps_max_layers_minus1, inclusive, when not present, the value of layer_id_in_nuh[i] is inferred to be equal to i.

When i is greater than 0, layer_id_in_nuh[i] shall be greater than layer_id_in_nuh[i – 1].

[Ed. (MH): When splitting_flag is equal to 1, the MSBs of layer_id_in_nuh should be required to be 0 if the total number of bits in segments is less than 6]

For i in a range from 0 to vps_max_layers_minus1, inclusive, the variable LayerIdxInVps[layer_id_in_nuh[i]] is set equal to i.

dimension_id[i][j] specifies the identifier of the j-th present scalability dimension type of the i-th layer. The number of bits used for the representation of dimension_id[i][j] is dimension_id_len_minus1[j] + 1 bits. When dimension_id[i][j] is not present for j in the range of 0 to NumScalabilityTypes − 1, inclusive, dimension_id[i][j] is inferred to be equal to ((layer_id_in_nuh[i] & ((1 << dimBitOffset[j + 1]) − 1)) >> dimBitOffset[j]).

The variable ScalabilityId[i][smIdx] specifying the identifier of the smIdx-th scalability dimension type of the i-th layer, the variable ViewId[layer_id_in_nuh[i]] specifying the view identifier of the i-th layer and the variable ViewScalExtLayerFlag specifying whether the i-th layer is a view scalability extension layer are derived as follows:

```
for ( i = 0; i  <=  vps_max_layers_minus1; i++) {
    lId = layer_id_in_nuh[ i ]
    for( smIdx= 0, j =0; smIdx < 16; smIdx ++ )
        if( scalability_mask[ smIdx ] )
            ScalabilityId[ i ][ smIdx ] = dimension_id[ i ][ j++ ]
    ViewId[ lId ] = ScalabilityId[ i ][ 0 ]
    ViewScalExtLayerFlag[ lId ] = ( ViewId[ lId ]  !=  ViewId[ 0 ] )
}
```

[Ed. (JB): Syntax and semantics not compatible with SHVC, or with use of splitting_flag.]

direct_dependency_flag[i][j] equal to 0 specifies that the layer with index j is not a direct reference layer for the layer with index i. direct_dependency_flag[i][j] equal to 1 specifies that the layer with index j may be a direct reference layer for the layer with index i. When direct_dependency_flag[i][j] is not present for i and j in the range of 0 to vps_max_layers_minus1, it is inferred to be equal to 0.

max_tid_il_ref_pics_plus1[i] equal to 0 specifies that within the CVS non-IRAP pictures with nuh_layer_id equal to layer_id_in_nuh[i] are not used as reference for inter-layer prediction. max_tid_il_ref_pics_plus1[i] greater than 0 specifies that within the CVS pictures with nuh_layer_id equal to layer_id_in_nuh[i] and TemporalId greater than max_tid_il_ref_pics_plus1[i] – 1 are not used as reference for inter-layer prediction. When not present, max_tid_il_ref_pics_plus1[i] is unspecified.

NOTE 1 – max_tid_il_ref_pics_plus1[i] equal to 7 does not impose a restriction on inter-layer prediction.

[Ed. (JB): Consider adding a presence flag to make it easy to opt out of imposing this restriction, and when not present, value would be inferred to be equal to 7.]

vps_number_layer_sets_minus1 plus 1 specifies the number of layer sets that are specified by the VPS. The value of vps_number_layer_sets_minus1 shall be in the range of 0 to 1023, inclusive, and shall be equal to vps_num_layer_sets_minus1.

vps_num_profile_tier_level_minus1 plus 1 specifies the number of profile_tier_level( ) syntax structures in the VPS.

vps_profile_present_flag[i] equal to 1 specifies that the profile and tier information for layer set i is present in the i-th profile_tier_level( ) syntax structure. vps_profile_present_flag[i] equal to 0 specifies that profile and tier information is not present in the i-th profile_tier_level( ) syntax structure and is inferred.

profile_ref_minus1[i] specifies that the profile and tier information for the i-th profile_tier_level( ) syntax structure is inferred to be equal to the profile and tier information for the (profile _ref_minus1[i] + 1)-th layer set. The value of profile_ref_minus1[i] + 1 shall be less than i.

more_output_layer_sets_than_default_flag equal to 1 specifies that the number of output layer sets specified by the VPS is greater than vps_number_layer_sets_minus1 + 1. more_output_layer_sets_than_default_flag equal to 0 specifies that the number of output layer sets specified by the VPS is equal to vps_number_layer_sets_minus1 + 1.

[Ed. (MH): The value of more_output_layer_sets_than_default_flag may be restricted to be equal to 0 by an SHVC profile, such that the number of output layer sets is equal to the number of layer sets.]

num_add_output_layer_sets_minus1 plus 1 specifies the number of output layer sets in addition to the default output layer sets specified by the VPS. The default output layer sets refer to the first vps_number_layer_sets_minus1 + 1 output layer sets specified by the VPS. For the default output layer sets, either only the highest layer is a target output layer or all layers are target output layers.

default_one_target_output_layer_flag equal to 1 specifies that only the highest layer in each of the default output layer sets is a target output layer. default_one_target_output_layer_flag equal to 0 specifies that all layers in each of the default output layer sets are target output layers.

[Ed. (MH): The value of default_one_target_output_layer_flag may be restricted to be equal to 1 by an SHVC profile and 0 by a MV-HEVC profile or a 3D-HEVC profile.]

output_layer_set_idx_minus1[i] plus 1 specifies the index of the layer set for the i-th output layer set. The value of output_layer_set_idx_minus1[i] shall be in the range of 0 to vps_num_layer_sets_minus1 − 1, inclusive. The length of the output_layer_set_idx_minus1[i] syntax element is Ceil(Log2(vps_num_layer_sets_minus1)) bits.

The layer set for the i-th output layer set with i in the range of 0 to vps_num_layer_sets_minus1, inclusive, is inferred to be the i-th layer set.

output_layer_flag[i][j] equal to 1 specifies that the j-th layer in the i-th output layer set is a target output layer. output_layer_flag[i][j] equal to 0 specifies that the j-th layer in the i-th output layer set is not a target output layer.

profile_level_tier_idx[i] specifies the index, into the list of profile_tier_level( ) syntax structures in the VPS, of the profile_tier_level( ) syntax structure that applies to i-th output layer set. The length of the profile_level_tier_idx[i] syntax element is Ceil(Log2(vps_num_profile_tier_level_minus1 + 1)) bits. The value of profile_level_tier_idx[0] is inferred to be equal to 0. The value of profile_level_tier_idx[i] shall be in the range of 0 to vps_num_profile_tier_level_minus1, inclusive.

max_one_active_ref_layer_flag equal to 1 specifies that at most one picture is used for inter-layer prediction for each picture in the CVS. max_one_active_ref_layer_flag equal to 0 specifies that more than one picture may be used for inter-layer prediction for each picture in the CVS.

[Ed. (MH): The value of max_one_active_ref_layer_flag may be restricted to be equal to 1, e.g. by an SHVC profile, such that up to one picture is used for inter-layer reference.]

direct_dep_type_len_minus2 plus 2 specifies the number of bits of the direct_dependency_type[i][j] syntax element. In bitstreams conforming to this version of this Specification the value of direct_dep_type_len_minus2 shall be equal 0. Although the value of direct_dep_type_len_minus2 shall be equal to 0 in this version of this Specification, decoders shall allow other values of direct_dep_type_len_minus2 in the range of 0 to 30, inclusive, to appear in the syntax.

direct_dependency_type[i][j] is used to derive the variables NumSamplePredRefLayers[i], NumMotionPredRefLayers[i], SamplePredEnabledFlag[i][j], and MotionPredEnabledFlag[i][j]. direct_dependency_type[i][j] shall be in the range of 0 to 2, inclusive, in bitstreams conforming to this version of this Specification. Although the value of direct_dependency_type[i][j] shall be in the range of 0 to 2, inclusive, in this version of this Specification, decoders shall allow values of direct_dependency_type[i][j] in the range of 3 to 2^{32}-2, inclusive, to appear in the syntax.

The variables NumSamplePredRefLayers[i], NumMotionPredRefLayers[i], SamplePredEnabledFlag[i][j], MotionPredEnabledFlag[i][j], NumDirectRefLayers[i], RefLayerId[i][j], MotionPredRefLayerId[i][j], and SamplePredRefLayerId[i][j] are derived as follows:

```
for( i = 0; i < 64; i++ ) {
    NumSamplePredRefLayers[ i ] = 0
    NumMotionPredRefLayers[ i ] = 0
    NumDirectRefLayers[ i ] = 0
    for( j = 0; j < 64; j++ ) {
        SamplePredEnabledFlag[ i ][ j ] = 0
        MotionPredEnabledFlag[ i ][ j ] = 0
        RefLayerId[ i ][ j ] = 0
        SamplePredRefLayerId[ i ][ j ] = 0
        MotionPredRefLayerId[ i ][ j ] = 0
    }
}

for( i = 1; i  <=  vps_max_layers_minus1; i++ ) {
    iNuhLId = layer_id_in_nuh[ i ]
```

```
        for( j = 0; j < i; j++ )
            if( direct_dependency_flag[ i ][ j ] ) {
                RefLayerId[ iNuhLId ][ NumDirectRefLayers[ iNuhLId ]++ ] = layer_id_in_nuh[ j ]
                SamplePredEnabledFlag[ iNuhLId ][ j ] = ( ( direct_dependency_type[ i ][ j ] + 1 ) & 1 )
                NumSamplePredRefLayers[ iNuhLId ] += SamplePredEnabledFlag[ iNuhLId ][ j ]
                MotionPredEnabledFlag[ iNuhLId ][ j ] = ( ( ( direct_dependency_type[ i ][ j ] + 1 ) & 2 ) >> 1 )
                NumMotionPredRefLayers[ iNuhLId ] += MotionPredEnabledFlag[ iNuhLId ][ j ]
            }
    }

    for( i = 1, mIdx = 0, sIdx = 0; i <= vps_max_layers_minus1; i++ ) {
        iNuhLId = layer_id_in_nuh[ i ]
        for( j = 0, j < i; j++ ) {
            if( MotionPredEnabledFlag[ iNuhLId ][ j ] )
                MotionPredRefLayerId[ iNuhLId ][ mIdx++ ] = layer_id_in_nuh[ j ]
            if( SamplePredEnabledFlag[ INuhLid ][ j ] )
                SamplePredRefLayerId[ iNuhLid ][ sIdx++ ] = layer_id_in_nuh[ j ]
        }
    }
```

When avc_base_layer_flag is equal to 1, it is a requirement of bitstream conformance that MotionPredRefLayerId[iNuhLId][mIdx] shall not be equal to 0 for iNuhLId equal to any value of nuh_layer_id present in the bitstream and any value of mIdx in the range of 0 to NumMotionPredRefLayers[iNuhLId] – 1, inclusive.

vps_shvc_reserved_zero_flag shall be equal to 0 in bitstreams conforming to this version of this Specification. Other values for vps_shvc_reserved_zero_flag are reserved for future use by ITU-T | ISO/IEC. Although the value of vps_shvc_reserved_zero_flag is required to be equal to 0 in this version of this Specification, decoders shall allow other values of vps_shvc_reserved_zero_flag to appear in the syntax.

NOTE 2 – It is anticipated that in future scalable extensions of this Specification, this field will be used to indicate either that all the VCL NAL units of an access unit have the same nuh_layer_id value or that two nuh_layer_id values are used by the VCL NAL units of an access unit and the picture with the greater nuh_layer_id value is an IRAP picture.

[Ed. (GT): vps_shvc_reserved_zero_flag corresponds to single_layer_for_non_irap_flag in SHVC draft.]

F.7.4.3.2 Sequence parameter set RBSP semantics

The specifications in subclause 7.4.3.2 apply, with following additions and modifications.

sps_max_sub_layers_minus1 plus 1 specifies the maximum number of temporal sub-layers that may be present in each CVS referring to the SPS. The value of sps_max_sub_layers_minus1 shall be in the range of 0 to 6, inclusive. When not present sps_max_sub_layers_minus1 is inferred to be equal to vps_max_sub_layers_minus1.

sps_temporal_id_nesting_flag, when sps_max_sub_layers_minus1 is greater than 0, specifies whether inter prediction is additionally restricted for CVSs referring to the SPS. When vps_temporal_id_nesting_flag is equal to 1, sps_temporal_id_nesting_flag shall be equal to 1. When sps_max_sub_layers_minus1 is equal to 0, sps_temporal_id_nesting_flag shall be equal to 1. When not present sps_temporal_id_nesting_flag is inferred to be equal to vps_temporal_id_nesting_flag.

NOTE 3 – The syntax element sps_temporal_id_nesting_flag is used to indicate that temporal up-switching, i.e. switching from decoding up to any TemporalId tIdN to decoding up to any TemporalId tIdM that is greater than tIdN, is always possible in the CVS.

sps_extension_flag equal to 0 specifies that no sps_extension( ) syntax structure is present in the SPS RBSP syntax structure. sps_extension_flag equal to 1 specifies that the sps_extension( ) syntax structure is present in the SPS RBSP syntax structure.

sps_extension2_flag equal to 0 specifies that no sps_extension_data_flag syntax elements are present in the SPS RBSP syntax structure. sps_extension2_flag shall be equal to 0 in bitstreams conforming to this version of this Specification. The value of 1 for sps_extension2_flag is reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore all sps_extension_data_flag syntax elements that follow the value 1 for sps_extension2_flag in an SPS NAL unit.

F.7.4.3.2.1 Sequence parameter set extension semantics

inter_view_mv_vert_constraint_flag equal to 1 specifies that vertical component of motion vectors used for inter-layer prediction are constrained in the CVS. When inter_view_mv_vert_constraint_flag is equal to 1, the vertical component of the motion vectors used for inter-layer prediction shall be equal to or less than 56 in units of luma samples. When inter_view_mv_vert_constraint_flag is equal to 0, no constraint for of the vertical component of the motion vectors used for inter-layer prediction is signalled by this flag. When not present, the inter_view_mv_vert_constraint_flag is inferred to be equal to 0.

sps_shvc_reserved_zero_idc shall be equal to 0 in bitstreams conforming to this version of this Specification. Other values for sps_shvc_reserved_zero_flag are reserved for future use by ITU-T | ISO/IEC. Although the value of sps_shvc_reserved_zero_flag is required to be equal to 0 in this version of this Specification, decoders shall allow other values of sps_shvc_reserved_zero_flag to appear in the syntax.

NOTE 4 – It is anticipated that in future scalable extensions of this Specification, this field will be used to indicate that the scaled reference layer offset parameters are present in the SPS.

[Ed. (GT): sps_shvc_reserved_zero_flag corresponds to scaled_ref_layer_offset_present_flag in the SHVC draft.]

F.7.4.3.3 Picture parameter set RBSP semantics

The specifications in subclause 7.4.3.3 apply, with the following modifications:

num_extra_slice_header_bits specifies the number of extra slice header bits that are present in the slice header RBSP for coded pictures referring to the PPS. num_extra_slice_header_bits shall be equal to 0 or 1 in bitstreams conforming to this version of this Specification. Other values for num_extra_slice_header_bits are reserved for future use by ITU-T | ISO/IEC. However, decoders shall allow num_extra_slice_header_bits to have any value.

F.7.4.3.4 Supplemental enhancement information RBSP semantics

The specifications in subclause 7.4.3.4 apply.

F.7.4.3.5 Access unit delimiter RBSP semantics

The specifications in subclause 7.4.3.5 apply.

F.7.4.3.6 End of sequence RBSP semantics

The specifications in subclause 7.4.3.6 apply.

F.7.4.3.7 End of bitstream RBSP semantics

The specifications in subclause 7.4.3.7 apply.

F.7.4.3.8 Filler data RBSP semantics

The specifications in subclause 7.4.3.8 apply.

F.7.4.3.9 Slice segment layer RBSP semantics

The specifications in subclause 7.4.3.9 apply.

F.7.4.3.10 RBSP slice segment trailing bits semantics

The specifications in subclause 7.4.3.10 apply.

F.7.4.3.11 RBSP trailing bits semantics

The specifications in subclause 7.4.3.11 apply.

F.7.4.3.12 Byte alignment semantics

The specifications in subclause 7.4.3.12 apply.

F.7.4.4 Profile, tier and level semantics

The profile_tier_level( ) syntax structure provides profile, tier and level information used for a layer set. When the profile_tier_level( ) syntax structure is included in a vps_extension( ) syntax structure, the applicable layer set to which the profile_tier_level( ) syntax structure applies is specified by the corresponding lsIdx variable in the vps_extension( ) syntax structure. When the profile_tier_level( ) syntax structure is included in a VPS, but not in a vps_extension( ) syntax structure, the applicable layer set to which the profile_tier_level( ) syntax structure applies is the layer set specified by the index 0. When the profile_tier_level( ) syntax structure is included in an SPS, the layer set to which the profile_tier_level( ) syntax structure applies is the layer set specified by the index 0.

For interpretation of the following semantics, CVS refers to the CVS subset associated with the layer set to which the profile_tier_level( ) syntax structure applies.

When the syntax elements general_profile_space, general_tier_flag, general_profile_idc, general_profile_compatibility_flag[j], general_progressive_source_flag, general_interlaced_source_flag, general_non_packed_constraint_flag, general_frame_only_constraint_flag, general_reserved_zero_44bits are not present for the applicable layer set, they are inferred to be equal to the corresponding values of the layer set specified by the index (profile_layer_set_ref_minus1[lsIdx] +1).

When the syntax elements sub_layer_profile_space[i], sub_layer_tier_flag[i], sub_layer_profile_idc[i], sub_layer_profile_compatibility_flag[i][j], sub_layer_progressive_source_flag[i], sub_layer_interlaced_source_flag[i], sub_layer_non_packed_constraint_flag[i], sub_layer_frame_only_constraint_flag[i], sub_layer_reserved_zero_44bits[i] are not present for the applicable layer set, and they are present in or inferred for the layer set specified by the index (profile_layer_set_ref_minus1[lsIdx] +1) they are inferred to be equal to the corresponding values of the layer set specified by the index (profile_layer_set_ref_minus1[lsIdx] +1).

The specifications in subclause 7.4.4 apply, with following modifications.

general_tier_flag specifies the tier context for the interpretation of general_level_idc as specified in Annex A or subclause G.11.

general_profile_idc, when general_profile_space is equal to 0, indicates a profile to which the CVS conforms as specified in Annex A or in subclause G.11. Bitstreams shall not contain values of general_profile_idc other than those specified in Annex A or subclause G.11. Other values of general_profile_idc are reserved for future use by ITU-T | ISO/IEC.

general_profile_compatibility_flag[j] equal to 1, when general_profile_space is equal to 0, indicates that the CVS conforms to the profile indicated by general_profile_idc equal to i as specified in Annex A or in subclause G.11. When general_profile_space is equal to 0, general_profile_compatibility_flag[general_profile_idc] shall be equal to 1. The value of general_profile_compatibility_flag[j] shall be equal to 0 for any value of j that is not specified as an allowed value of general_profile_idc in Annex A or in subclause G.11.

general_level_idc indicates a level to which the CVS conforms as specified in Annex A or subclause G.11. Bitstreams shall not contain values of general_level_idc other than those specified in Annex A or subclause G.11. Other values of general_level_idc are reserved for future use by ITU-T | ISO/IEC.

sub_layer_profile_present_flag[i] equal to 1, specifies that profile information is present in the profile_tier_level( ) syntax structure for the representation of the sub-layer with TemporalId equal to i. sub_layer_profile_present_flag[i] equal to 0 specifies that profile information is not present in the profile_tier_level( ) syntax structure for the representations of the sub-layer with TemporalId equal to i. When profilePresentFlag is equal to 0, sub_layer_profile_present_flag[i] shall be equal to 0.

F.7.4.5 Scaling list data semantics

The specifications in subclause 7.4.5 apply

F.7.4.6 Supplemental enhancement information message semantics

The specifications in subclause 7.4.6 apply

F.7.4.7 Slice segment header semantics

F.7.4.7.1 General slice segment header semantics

The specifications in subclause 7.4.7.1 apply with the following modifications.

– "When nal_unit_type has a value in the range of 16 to 23, inclusive (IRAP picture), slice_type shall be equal to 2." is replaced by "When nal_unit_type has a value in the range of 16 to 23 and nuh_layer_id is equal to 0, inclusive (IRAP picture), slice_type shall be equal to 2."

discardable_flag equal to 1 specifies that the coded picture is not used as a reference picture for inter prediction and is not used as an inter-layer reference picture in the decoding process of subsequent pictures in decoding order. discardable_flag equal to 0 specifies that the coded picture may be used as a reference picture for inter prediction and may be used as an inter-layer reference picture in the decoding process of subsequent pictures in decoding order. When not present, the value of discardable_flag is inferred to be equal to 0.

inter_layer_pred_enabled_flag equal to 1 specifies that inter-layer prediction may be used in decoding of the current picture. inter_layer_pred_enabled_flag equal to 0 specifies that inter-layer prediction is not used in decoding of the current picture. When not present, the value of inter_layer_pred_enabled_flag is inferred to be equal to 0.

num_inter_layer_ref_pics_minus1 plus 1 specifies the number of pictures that may be used in decoding of the current picture for inter-layer prediction. The length of the num_inter_layer_ref_pics_minus1 syntax element is Ceil(Log2(NumDirectRefLayers[nuh_layer_id])) bits. The value of num_inter_layer_ref_pics_minus1 shall be in the range of 0 to NumDirectRefLayers[nuh_layer_id] − 1, inclusive.

The variable NumActiveRefLayerPics is derived as follows:

```
if( nuh_layer_id == 0 || NumDirectRefLayers[ nuh_layer_id ] == 0 || !inter_layer_pred_enabled_flag )
    NumActiveRefLayerPics = 0
else if( max_one_active_ref_layer_flag || NumDirectRefLayers[ nuh_layer_id ] = = 1 )
    NumActiveRefLayerPics = 1
else
    NumActiveRefLayerPics = num_inter_layer_ref_pics_minus1 + 1
```

All slices of a coded picture shall have the same value of NumActiveRefLayerPics.

inter_layer_pred_layer_idc[i] specifies the variable, RefPicLayerId[i], representing the nuh_layer_id of the i-th picture that may be used by the current picture for inter-layer prediction. The length of the syntax element inter_layer_pred_layer_idc[i] is Ceil(Log2(NumDirectRefLayers[nuh_layer_id])) bits. The value of inter_layer_pred_layer_idc[i] shall be in the range of 0 to NumDirectRefLayers[nuh_layer_id] − 1, inclusive. When not present, the value of inter_layer_pred_layer_idc[i] is inferred to be equal to 0.

When i is greater than 0, inter_layer_pred_layer_idc[i] shall be greater than inter_layer_pred_layer_idc[i − 1]. [Ed. (JB): This restriction was imposed to make behavior match the earlier design of increasing entries, but is an area noted for future study.]

The variables RefPicLayerId[i] for each value of i in the range of 0 to NumActiveRefLayerPics − 1, inclusive, NumActiveMotionPredRefLayers, and ActiveMotionPredRefLayerId[j] for each value of j in the range of 0 to NumActiveMotionPredRefLayers − 1, inclusive, are derived as follows:

```
for( i = 0, j = 0; i < NumActiveRefLayerPics; i++)
    RefPicLayerId[ i ] = RefLayerId[ nuh_layer_id ][ inter_layer_pred_layer_idc[ i ] ]
    if( MotionPredEnabledFlag[ nuh_layer_id ][ inter_layer_pred_layer_idc[ i ] ] )
        ActiveMotionPredRefLayerId[ j++ ] = RefLayerId[ nuh_layer_id ][ inter_layer_pred_layer_idc[ i ] ]
}
NumActiveMotionPredRefLayers = j
```

All slices of a picture shall have the same value of inter_layer_pred_layer_idc[i] for each value of i in the range of 0 to NumActiveRefLayerPics − 1, inclusive.

It is a requirement of bitstream conformance that for each value of i in the range of 0 to NumActiveRefLayerPics − 1, inclusive, either of the following two conditions shall be true:

– The value of max_tid_il_ref_pics_plus1[LayerIdxInVps[RefPicLayerId[i]]] is greater than TemporalId.

– The values of max_tid_il_ref_pics_plus1[LayerIdxInVps[RefPicLayerId[i]]] and TemporalId are both equal to 0 and the picture in the current access unit with nuh_layer_id equal to RefPicLayerId[i] is an IRAP picture.

It is a requirement of bitstream conformance that for each value of i in the range of 0 to NumActiveRefLayerPics − 1, inclusive, the value of SamplePredEnabledFlag[nuh_layer_id] [RefPicLayerId[i]] or MotionPredEnabledFlag[nuh_layer_id] [RefPicLayerId[i]] shall be equal to 1.

[Ed. (JB): In future extensions, the above requirement may be changed. (YK): Just to understand: in which scenarios this requirement may be changed? (MH): If a new inter-layer prediction type is introduced in a future extension, there may be a reference layer which is only used as reference for this new inter-layer prediction type but not for inter-layer sample or motion prediction.]

inter_layer_sample_pred_only_flag equal to 1 specifies that inter prediction is not used in decoding of the current picture. inter_layer_sample_pred_only_flag equal to 0 specifies that inter prediction may be used in decoding of the current picture. When not present, the value of inter_layer_sample_pred_only_flag is inferred to be equal to 0.

The variable InterRefEnabledInRPLFlag is derived as follows:.

– If NumSamplePredRefLayers[nuh_layer_id] is greater than 0 and NumActiveRefLayerPics is greater than 0, InterRefEnabledInRPLFlag is set equal to !inter_layer_sample_pred_only_flag.

– Otherwise, InterRefEnabledInRPLFlag is set equal to 1

alt_collocated_indication_flag equal to 0 specifies that a collocated picture for temporal motion vector prediction is indicated by collocated_from_l0_flag, when present, and collocated_ref_idx. alt_collocated_indication_flag equal to 1 specifies that a collocated picture for temporal motion vector prediction is indicated by collocated_ref_layer_idx. When alt_collocated_indication_flag is not present, it is inferred to be equal to 0.

It is a requirement of bitstream conformance that the value of alt_collocated_indication_flag shall be the same for all slices of a coded picture.

collocated_ref_layer_idx specifies the collocated picture for temporal motion vector prediction as specified in subclause 8.5.3.2.7. When alt_collocated_indication_flag is equal to 1 and NumActiveMotionPredRefLayers is equal to 1, collocated_ref_layer_idx is inferred to be equal to 0. collocated_ref_layer_idx shall be in the range of 0 to NumActiveMotionPredRefLayers – 1, inclusive.

It is a requirement of bitstream conformance that the picture referred to by collocated_ref_layer_idx shall be the same for all slices of a coded picture..

F.7.4.7.2 Reference picture list modification semantics

The specifications in subclause 7.4.7.2 apply with following modifications.

– Equation (7-43) specifying the derivation of NumPocTotalCurr is replaced by:

[Ed. (YK): With the addition of the inter-layer stuff, this variable name gets confusing. Maybe it should be changed to "NumPicTotalCurr". Purely editorial - can be done later on, even in Version 1.]

```
NumPocTotalCurr = 0
for( i = 0; i < NumNegativePics[ CurrRpsIdx ]; i++)
  if(UsedByCurrPicS0[ CurrRpsIdx ][ i ] )
      NumPocTotalCurr++
for( i = 0; i < NumPositivePics[ CurrRpsIdx ]; i++)  (G-1)
  if(UsedByCurrPicS1[ CurrRpsIdx ][ i ] )
      NumPocTotalCurr++
for( i = 0; i < num_long_term_sps + num_long_term_pics; i++ )
  if( UsedByCurrPicLt[ i ] )
      NumPocTotalCurr++
NumPocTotalCurr += NumActiveRefLayerPics
```

F.7.4.7.3 Weighted prediction parameters semantics

The specifications in subclause 7.4.7.3 apply

F.7.4.8 Short-term reference picture set semantics

The specifications in subclause 7.4.8 apply

F.7.4.9 Slice segment data semantics

F.7.4.9.1 General slice segment data semantics

The specifications in subclause 7.4.9.1 apply.

F.7.4.9.2 Coding tree unit semantics

The specifications in subclause 7.4.9.2 apply.

F.7.4.9.3 Sample adaptive offset semantics

The specifications in subclause 7.4.9.3 apply.

F.7.4.9.4 Coding quadtree semantics

The specifications in subclause 7.4.9.4 apply.

F.7.4.9.5 Coding unit semantics

The specifications in subclause 7.4.9.5 apply.

F.7.4.9.6 Prediction unit semantics

The specifications in subclause 7.4.9.6 apply.

F.7.4.9.7 PCM sample semantics

The specifications in subclause 7.4.9.7 apply.

F.7.4.9.8 Transform tree semantics

The specifications in subclause 7.4.9.8 apply.

F.7.4.9.9 Motion vector difference semantics

The specifications in subclause 7.4.9.9 apply.

F.7.4.9.10 Transform unit semantics

The specifications in subclause 7.4.9.10 apply.

F.7.4.9.11 Residual coding semantics

The specifications in subclause 7.4.9.11 apply.

F.8 Decoding process

F.8.1 General decoding process

The specifications in subclause 8.1 apply with following additions.

When the current picture has nuh_layer_id greater than 0, the following applies.

– Depending on the value of separate_colour_plane_flag, the decoding process is structured as follows:

  – If separate_colour_plane_flag is equal to 0, the following decoding process is invoked a single time with the current picture being the output.

  – Otherwise (separate_colour_plane_flag is equal to 1), the following decoding process is invoked three times. Inputs to the decoding process are all NAL units of the coded picture with identical value of colour_plane_id. The decoding process of NAL units with a particular value of colour_plane_id is specified as if only a CVS with monochrome colour format with that particular value of colour_plane_id would be present in the bitstream. The output of each of the three decoding processes is assigned to one of the 3 sample arrays of the current picture, with the NAL units with colour_plane_id equal to 0, 1 and 2 being assigned to S_L, S_{Cb}, and S_{Cr}, respectively.

    NOTE 5 – The variable ChromaArrayType is derived as equal to 0 when separate_colour_plane_flag is equal to 1 and chroma_format_idc is equal to 3. In the decoding process, the value of this variable is evaluated resulting in operations identical to that of monochrome pictures (when chroma_format_idc is equal to 0).

– The decoding process operates as follows for the current picture CurrPic.

  – For the decoding of the slice segment header of the first slice, in decoding order, of the current picture, the decoding process for starting the decoding of a coded picture with nuh_layer_id greater than 0 specified in subclause F.8.1.1 is invoked.

  – When ViewScalExtLayerFlag[nuh_layer_id] is equal to 1, the decoding process for a coded picture with nuh_layer_id greater than 0 specified in subclause G.8.1 is invoked. [Ed. (MH): In future scalable extensions, other types of scalability identifiers may be examined here, and appropriate decoding processes invoked for them.]

  – After all slices of the current picture have been decoded, the decoding process for ending the decoding of a coded picture with nuh_layer_id greater than 0 specified in subclause F.8.1.2 is invoked.

F.8.1.1 Decoding process for starting the decoding of a coded picture with nuh_layer_id greater than 0

Each picture referred to in this subclause is a complete coded picture.

The decoding process operates as follows for the current picture CurrPic:

1. The decoding of NAL units is specified in subclause 4.
2. The processes in subclause 8.3 specify the following decoding processes using syntax elements in the slice segment layer and above:

   – Variables and functions relating to picture order count are derived in subclause 8.3.1. This needs to be invoked only for the first slice segment of a picture. It is a requirement of bitstream conformance that PicOrderCntVal shall remain unchanged within an access unit.

   – The decoding process for RPS in subclause 8.3.2 is invoked, wherein only reference pictures with a nuh_layer_id equal to that of CurrPic may be marked as "unused for reference" or "used for long-term reference" and any picture with a different value of nuh_layer_id is not marked. This needs to be invoked only for the first slice segment of a picture.

– When CurrPic is a BLA picture or is a CRA picture with NoRaslOutputFlag equal to 1, the decoding process for generating unavailable reference pictures specified in subclause 8.3.3 is invoked, which needs to be invoked only for the first slice segment of a picture.

F.8.1.2 Decoding process for ending the decoding of a coded picture with nuh_layer_id greater than 0

PicOutputFlag is set as follows:

– If the current picture is a RASL picture and NoRaslOutputFlag of the associated IRAP picture is equal to 1, PicOutputFlag is set equal to 0.

– Otherwise, PicOutputFlag is set equal to pic_output_flag.

The following applies:

– If discardable_flag is equal to 1, the decoded picture is marked as "unused for reference".

[Ed. (JC): The syntax is not defined in this annex, it might be good to move syntax and semantics of “General slice segment header syntax” from Annex G to Annex F. (MH): The syntax of discardable_flag should be moved to clause 7, as it concerns also the base layer slices and must be specified unambiguously even if multiview or scalable extensions are not in use. Other changes in slice segment header must remain in Annex G, as they only relate to refIdx based multiview/scalable coding.(GT): How to resolve this issue should be part of the general discussion on how to restructure the annexes.]

– Otherwise, the decoded picture is marked as "used for short-term reference".

When TemporalId is equal to HighestTid, the marking process for sub-layer non-reference pictures not needed for inter-layer prediction specified in subclause F.8.1.2.1 is invoked with latestDecLayerId equal to nuh_layer_id as input.

F.8.1.2.1 Marking process for sub-layer non-reference pictures not needed for inter-layer prediction

Input to this process is:

– a nuh_layer_id value latestDecLayerId

Output of this process is:

– potentially updated marking as "unused for reference" for some decoded pictures

NOTE 6 – This process marks pictures that are not needed for inter or inter-layer prediction as "unused for reference". When TemporalId is less than HighestTid, the current picture may be used for reference in inter prediction and this process is not invoked.

The variables numTargetDecLayers, and latestDecIdx are derived as follows:

– numTargetDecLayers is set equal to the number of entries in TargetDecLayerIdList.

– latestDecIdx is set equal to the value of i for which TargetDecLayerIdList[i] is equal to latestDecLayerId.

For i in the range of 0 to latestDecIdx, inclusive, the following applies for marking of pictures as "unused for reference":

– Let currPic be the picture in the current access unit with nuh_layer_id equal to TargetDecLayerIdList[i].

– When currPic is marked as "used for reference" and is a sub-layer non-reference picture, the following applies:

  – The variable currTid is set equal to the value of TemporalId of currPic.

  – The variable remainingInterLayerReferencesFlag is derived as specified in the following:

```
remainingInterLayerReferencesFlag = 0
if ( currTid  <=  ( max_tid_il_ref_pics_plus1[ LayerIdxInVps[ TargetDecLayerIdList[ i ] ] ] –1 ) )
    for( j = latestDecIdx + 1; j < numTargetDecLayers; j++ )
        for( k = 0; k < NumDirectRefLayers[ TargetDecLayerIdList[ j ] ]; k++ )
            if( TargetDecLayerIdList[ i ]  = =  RefLayerId[ TargetDecLayerIdList[ j ] ][ k ] )
                remainingInterLayerReferencesFlag = 1
```

  – When remainingInterLayerReferenceFlag is equal to 0, currPic is marked as "unused for reference".

F.8.2 NAL unit decoding process

The specifications in subclause 8.2 apply.

F.8.3 Slice decoding processes

F.8.3.1 (void)

(void)

F.8.3.2 (void)

(void)

F.8.3.3 (void)

(void)

F.8.3.4 Decoding process for reference picture lists construction

This process is invoked at the beginning of the decoding process for each P or B slice.

Reference pictures are addressed through reference indices as specified in subclause 8.5.3.3.2. A reference index is an index into a reference picture list. When decoding a P slice, there is a single reference picture list RefPicList0. When decoding a B slice, there is a second independent reference picture list RefPicList1 in addition to RefPicList0.

At the beginning of the decoding process for each slice, the reference picture lists RefPicList0 and, for B slices, RefPicList1 are derived as follows:

The variable NumRpsCurrTempList0 is set equal to Max(num_ref_idx_l0_active_minus1 + 1, NumPocTotalCurr) and the list RefPicListTemp0 is constructed as follows:

```
rIdx = 0
while( rIdx < NumRpsCurrTempList0 ) {
    if( InterRefEnabledInRPLFlag ) {
        for( i = 0; i < NumPocStCurrBefore  &&  rIdx < NumRpsCurrTempList0; rIdx++, i++ )
            RefPicListTemp0[ rIdx ] = RefPicSetStCurrBefore[ i ]
        for( i = 0;  i < NumPocStCurrAfter  &&  rIdx < NumRpsCurrTempList0; rIdx++, i++ )          (F-2)
            RefPicListTemp0[ rIdx ] = RefPicSetStCurrAfter[ i ]
        for( i = 0; i < NumPocLtCurr  &&  rIdx < NumRpsCurrTempList0; rIdx++, i++ )
            RefPicListTemp0[ rIdx ] = RefPicSetLtCurr[ i ]
    }
    for( i = 0; i < NumActiveRefLayerPics; rIdx++, i++ )
        RefPicListTemp0[ rIdx ] = RefPicSetInterLayer[ i ]
}
```

The list RefPicList0 is constructed as follows:

```
for( rIdx = 0; rIdx  <=  num_ref_idx_l0_active_minus1; rIdx++)                                    (F-3)
    RefPicList0[ rIdx ] = ref_pic_list_modification_flag_l0 ? RefPicListTemp0[ list_entry_l0[ rIdx ] ] :
                                                              RefPicListTemp0[ rIdx ]
```

When the slice is a B slice, the variable NumRpsCurrTempList1 is set equal to Max(num_ref_idx_l1_active_minus1 + 1, NumPocTotalCurr) and the list RefPicListTemp1 is constructed as follows:

```
rIdx = 0
while( rIdx < NumRpsCurrTempList1 ) {
    if( InterRefEnabledInRPLFlag ) {
        for( i = 0; i < NumPocStCurrAfter  &&  rIdx < NumRpsCurrTempList1; rIdx++, i++ )
            RefPicListTemp1[ rIdx ] = RefPicSetStCurrAfter[ i ]
        for( i = 0;  i < NumPocStCurrBefore  &&  rIdx < NumRpsCurrTempList1; rIdx++, i++ )          (F-4)
            RefPicListTemp1[ rIdx ] = RefPicSetStCurrBefore[ i ]
        for( i = 0; i < NumPocLtCurr  &&  rIdx < NumRpsCurrTempList1; rIdx++, i++ )
            RefPicListTemp1[ rIdx ] = RefPicSetLtCurr[ i ]
    }
    for( i = 0; i< NumActiveRefLayerPics; rIdx++, i++ )
        RefPicListTemp1[ rIdx ] = RefPicSetInterLayer[ i ]
}
```

When the slice is a B slice, the list RefPicList1 is constructed as follows:

```
for( rIdx = 0; rIdx <= num_ref_idx_l1_active_minus1; rIdx++)                    (F-5)
    RefPicList1[ rIdx ] = ref_pic_list_modification_flag_l1 ? RefPicListTemp1[ list_entry_l1[ rIdx ] ] :
                                                              RefPicListTemp1[ rIdx ]
```

F.8.4 Decoding process for coding units coded in intra prediction mode

The specifications in subclause 8.4 apply.

F.8.5 Decoding process for coding units coded in inter prediction mode

The specifications in subclause 8.5 apply with the following modifications:

– In subclause 8.5 and all its subclauses the invocations of the process specified in subclause 8.5.3.2.7 are replaced by invocations of subclause F.8.5.1.

[Ed. (YK): Is there a way to make sure that this change is a high-level syntax only change?]

F.8.5.1 Derivation process for temporal luma motion vector prediction

Inputs to this process are:

– a luma location (xPb, yPb) specifying the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture,

– two variables nPbW and nPbH specifying the width and the height of the luma prediction block,

– a reference index refIdxLX, with X being 0 or 1.

Outputs of this process are:

– the motion vector prediction mvLXCol,

– the availability flag availableFlagLXCol.

The variable currPb specifies the current luma prediction block at luma location (xPb, yPb).

The variables mvLXCol and availableFlagLXCol are derived as follows:

– If slice_temporal_mvp_enabled_flag is equal to 0, both components of mvLXCol are set equal to 0 and availableFlagLXCol is set equal to 0.

– Otherwise, the following ordered steps apply:

1. Depending on the values of alt_collocated_indication_flag, collocated_ref_layer_idx, slice_type, collocated_from_l0_flag, and collocated_ref_idx, the variable colPic, specifying the collocated picture, is derived as follows:

   – If alt_collocated_indication_flag is equal to 1, colPic is set equal to the picture in the current access unit with nuh_layer_id equal to ActiveMotionPredRefLayerId[collocated_ref_layer_idx].

   – Otherwise, if slice_type is equal to B and collocated_from_l0_flag is equal to 0, colPic is set equal to RefPicList1[collocated_ref_idx].

   – Otherwise (slice_type is equal to B and collocated_from_l0_flag is equal to 1 or slice_type is equal to P), colPic is set equal to RefPicList0[collocated_ref_idx].

2. The bottom right collocated motion vector is derived as follows:

$$xColBr = xPb + nPbW \quad (F\text{-}6)$$

$$yColBr = yPb + nPbH \quad (F\text{-}7)$$

   – If yPb >> CtbLog2SizeY is equal to yColBr >> CtbLog2SizeY, yColBr is less than pic_height_in_luma_samples, and xColBr is less than pic_width_in_luma_samples, the following applies:

     – The variable colPb specifies the luma prediction block covering the modified location given by ((xColBr >> 4) << 4, (yColBr >> 4) << 4) inside the collocated picture specified by colPic.

– The luma location (xColPb, yColPb) is set equal to the top-left sample of the collocated luma prediction block specified by colPb relative to the top-left luma sample of the collocated picture specified by colPic.

– The derivation process for collocated motion vectors as specified in subclause 8.5.3.2.8 is invoked with currPb, colPic, colPb, (xColPb, yColPb), and refIdxLX as inputs, and the output is assigned to mvLXCol and availableFlagLXCol.

– Otherwise, both components of mvLXCol are set equal to 0 and availableFlagLXCol is set equal to 0.

3. When availableFlagLXCol is equal to 0, the central collocated motion vector is derived as follows:

$$xColCtr = xPb + (nPbW >> 1) \quad \text{(F-8)}$$

$$yColCtr = yPb + (nPbH >> 1) \quad \text{(F-9)}$$

– The variable colPb specifies the luma prediction block covering the modified location given by ((xColCtr >> 4) << 4, (yColCtr >> 4) << 4) inside the colPic.

– The luma location (xColPb, yColPb) is set equal to the top-left sample of the collocated luma prediction block specified by colPb relative to the top-left luma sample of the collocated picture specified by colPic.

– The derivation process for collocated motion vectors as specified in subclause 8.5.3.2.8 is invoked with currPb, colPic, colPb, (xColPb, yColPb), and refIdxLX as inputs, and the output is assigned to mvLXCol and availableFlagLXCol.

F.8.6 Scaling, transformation and array construction process prior to deblocking filter process

The specifications in subclause 8.6 apply.

F.8.7 In-loop filter process

The specifications in subclause 8.7 apply.

F.9 Parsing process

The specifications in clause 9 apply.

F.10 Specification of bitstream subsets

The specifications in clause 10 apply.

F.11 (Void)

(void) [Ed. (MH): no profiles are intended to be specified as part of this annex, as it only specifies common syntax and semantics and some decoding processes for multiview extension and potential future scalable extensions.]

F.12 Byte stream format

The specifications in Annex B apply.

F.13 Hypothetical reference decoder

F.13.1 General

The specifications in subclause C.1 apply.

F.13.2 Operation of coded picture buffer (CPB)

The specifications in subclause C.2 apply with the following modifications. [Ed. (MH): Consider including the full text of C.2 here to improve readability.]

– Replace "a BLA access unit for which the coded picture has nal_unit_type equal to BLA_W_RADL or BLA_N_LP" with "a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_RADL or BLA_N_LP".

– Replace "a BLA access unit for which the coded picture has nal_unit_type equal to BLA_W_LP" with "a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_LP".

– Replace "picture n" with "access unit n".

– Replace "AuNominalRemovalTime[prevNonDiscardablePic] is the nominal removal time of the preceding picture in decoding order with TemporalId equal to 0 that is not a RASL, RADL or sub-layer non-reference picture", with "AuNominalRemovalTime[prevNonDiscardablePic] is the nominal removal time of the preceding access unit in decoding order, each picture of which is with TemporalId equal to 0 that is not a RASL, RADL or sub-layer non-reference picture".

F.13.3 Operation of the decoded picture buffer (DPB)

F.13.3.1 General

[Ed. (MH/GT): It is for further study whether the DPB operates a) independently for each layer, b) independently for pictures with the same spatial resolution, c) jointly for all pictures across different layers.]

The specifications in this subclause apply independently to each set of DPB parameters selected as specified in subclause C.1.

The decoded picture buffer contains picture storage buffers. Each of the picture storage buffers may contain a decoded picture that is marked as "used for reference" or is held for future output. The processes specified in subclauses F.13.3.2, F.13.3.3 and F.13.3.4 are sequentially applied as specified below.

PicOutputFlag for pictures that are not included in a target output layer is set equal to 0.

Decoded pictures with the same DPB output time and with PicOutputFlag equal to 1 are output in ascending order of nuh_layer_id values of these decoded pictures.

Let picture n be the coded picture or decoded picture of the access unit n for a particular value of nuh_layer_id, wherein n is a non-negative integer number. [Ed. (CY&YK): This probably is not a good definition of picture n especially if each picture is a DU. It is a temporary term defined only for DPB operations, further improvements are needed.]

F.13.3.2 Removal of pictures from the DPB

The specifications in subclause C.3.2 apply separately for each set of decoded pictures with a particular value of nuh_layer_id with the following modifications. [Ed. (CY): need to unify active SPS and active layer SPS, otherwise, the text in C.3.2 needs to be further extended to accommodate the case when the SPS is an active layer SPS.]

– Replace "The removal of pictures from the DPB before decoding of the current picture (but after parsing the slice header of the first slice of the current picture) happens instantaneously at the CPB removal time of the first decoding unit of access unit n (containing the current picture) and proceeds as follows:" with "The removal of pictures from the DPB before decoding of the current picture (but after parsing the slice header of the first slice of the current picture) happens instantaneously at the CPB removal time of the first decoding unit of the picture n and proceeds as follows:".

F.13.3.3 Picture output

The specifications in subclause C.3.3 apply with the following modifications.

– Replace "The output of the current picture is specified as follows, " with "For each picture of the current access unit, the output of the picture is specified as follows for each picture of the acces unit ".

F.13.3.4 Current decoded picture marking and storage

The process specified in this subclause happens instantaneously at the CPB removal time of the last decoding unit of picture n, CpbRemovalTime[n].

The current decoded picture is stored in the DPB in an empty picture storage buffer, the DPB fullness is incremented by one, and the current picture is marked as "used for short-term reference".

F.13.4 Bitstream conformance

A bitstream of coded data conforming to this Specification shall fulfil all requirements specified in this subclause.

The bitstream shall be constructed according to the syntax, semantics, and constraints specified in this Specification outside of this annex.

The first access unit in a bitstream shall be an IRAP access unit, i.e. an IDR access unit, a CRA access unit or a BLA access unit.

The bitstream is tested by the HRD for conformance as specified in subclause C.1.

Let the nuh_layer_id of the current picture be currPicLayerId.

For each current picture, let the variables maxPicOrderCnt and minPicOrderCnt be set equal to the maximum and the minimum, respectively, of the PicOrderCntVal values of the following pictures with nuh_layer_id equal to currPicLayerId:

– The current picture.

– The previous picture in decoding order that has TemporalId equal to 0 and that is not a RASL picture, a RADL picture, or a sub-layer non-reference picture.

– The short-term reference pictures in the RPS of the current picture.

– All pictures n that have PicOutputFlag equal to 1, AuCpbRemovalTime[n] less than AuCpbRemovalTime[currPic], and DpbOutputTime[n] greater than or equal to AuCpbRemovalTime[currPic], where currPic is the current picture. [Ed. (CY): clarify the AuCpbRemovalTime of a picture to be that of the containing AU.]

All of the following conditions shall be fulfilled for each of the bitstream conformance tests:

1. For each access unit n, with n greater than 0, associated with a buffering period SEI message, let the variable deltaTime90k[n] be specified as follows:

   deltaTime90k[n] = 90000 * (AuNominalRemovalTime[n] − AuFinalArrivalTime[n − 1]) (F-10)

   The value of InitCpbRemovalDelay[SchedSelIdx] is constrained as follows:

   – If cbr_flag[SchedSelIdx] is equal to 0, the following condition shall be true:

     InitCpbRemovalDelay[SchedSelIdx] <= Ceil(deltaTime90k[n]) (F-11)

   – Otherwise (cbr_flag[SchedSelIdx] is equal to 1), the following condition shall be true:

     Floor(deltaTime90k[n]) <= InitCpbRemovalDelay[SchedSelIdx] <= Ceil(deltaTime90k[n]) (F-12)

     NOTE 1 – The exact number of bits in the CPB at the removal time of each picture may depend on which buffering period SEI message is selected to initialize the HRD. Encoders must take this into account to ensure that all specified constraints must be obeyed regardless of which buffering period SEI message is selected to initialize the HRD, as the HRD may be initialized at any one of the buffering period SEI messages.

2. A CPB overflow is specified as the condition in which the total number of bits in the CPB is greater than the CPB size. The CPB shall never overflow.

3. A CPB underflow is specified as the condition in which the nominal CPB removal time of decoding unit m DuNominalRemovalTime(m) is less than the final CPB arrival time of decoding unit m DuFinalArrivalTime(m) for at least one value of m. When low_delay_hrd_flag[HighestTid] is equal to 0, the CPB shall never underflow.

4. When SubPicHrdFlag is equal to 1, low_delay_hrd_flag[HighestTid] is equal to 1, and the nominal removal time of a decoding unit m of access unit n is less than the final CPB arrival time of decoding unit m (i.e. DuNominalRemovalTime[m] < DuFinalArrivalTime[m]), the nominal removal time of access unit n shall be less than the final CPB arrival time of access unit n (i.e. AuNominalRemovalTime[n] < AuFinalArrivalTime[n]).

5. The nominal removal times of access units from the CPB (starting from the second access unit in decoding order) shall satisfy the constraints on AuNominalRemovalTime[n] and AuCpbRemovalTime[n] expressed in subclauses A.4.1 through A.4.2.

6. For each current picture, after invocation of the process for removal of pictures from the DPB as specified in subclause 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] less than AuCpbRemovalTime[currPic], where currPic is the current picture, shall be less than or equal to sps_max_dec_pic_buffering_minus1[HighestTid]. [Ed. (MH): This constaint should be made nuh_layer_id specific similarly to the DPB constraints below, i.e. “sps_max_dec_pic_buffering_minus1[HighestTid] of the active SPS (when currLayerId is equal to 0) or the active layer SPS for the value of currLayerId (when currLayerId is not equal to 0).]

7. All reference pictures shall be present in the DPB when needed for prediction. Each picture that has PicOutputFlag equal to 1 shall be present in the DPB at its DPB output time unless it is removed from the DPB before its output time by one of the processes specified in subclause C.3.

8. For each current picture, the value of maxPicOrderCnt − minPicOrderCnt shall be less than MaxPicOrderCntLsb / 2.

9. The value of DpbOutputInterval[n] as given by Equation C-17, which is the difference between the output time of an access uint and that of the first access unit following it in output order and having PicOutputFlag equal to 1, shall satisfy the constraint expressed in subclause A.4.1 for the profile, tier and level specified in the bitstream using the decoding process specified in clauses 2 through 10. [Ed. (MH): This constraint has to be updated, since 1) it assumes a single profile-tier-level combination for a bitstream (as if the bitstream were a single-layer bitstream), and 2) it refers to the decoding process in clauses 2 to 10 (while now also the decoding process of extensions should somehow be referred to).]

10. For each current picture, when sub_pic_cpb_params_in_pic_timing_sei_flag is equal to 1, let tmpCpbRemovalDelaySum be derived as follows:

```
tmpCpbRemovalDelaySum = 0
for( i = 0; i < num_decoding_units_minus1; i++ )                    (F-13)
    tmpCpbRemovalDelaySum  +=  du_cpb_removal_delay_increment_minus1[ i ] + 1
```

The value of ClockSubTick * tmpCpbRemovalDelaySum shall be equal to the difference between the nominal CPB removal time of the current access unit and the nominal CPB removal time of the first decoding unit in the current access unit in decoding order.

F.13.5 Decoder conformance

F.13.5.1 General

The specifications in subclause C.5.1 apply.

F.13.5.2 Operation of the output order DPB

F.13.5.2.1 General

The decoded picture buffer contains picture storage buffers. The number of picture storage buffers for nuh_layer_id equal to 0 is derived from the active SPS. The number of picture storage buffers for each non-zero nuh_layer_id value is derived from the active layer SPS for that non-zero nuh_layer_id value. Each of the picture storage buffers contains a decoded picture that is marked as "used for reference" or is held for future output. The process for output and removal of pictures from the DPB as specified in subclause F.13.5.2.2 is invoked, followed by the invocation of the process for picture decoding, marking, additional bumping, and storage as specified in subclause F.13.5.2.3. The "bumping" process is specified in subclause F.13.5.2.4 and is invoked as specified in subclauses F.13.5.2.2 and F.13.5.2.3.

Let picture n be the coded picture or decoded picture of the access unit n for a particular value of nuh_layer_id, wherein n is a non-negative integer number.

F.13.5.2.2 Output and removal of pictures from the DPB

The output and removal of pictures from the DPB before the decoding of the current picture (but after parsing the slice header of the first slice of the current picture) happens instantaneously when the first decoding unit of the current picture is removed from the CPB and proceeds as follows:

The decoding process for RPS as specified in subclause 8.3.2 is invoked. [Ed. (CY): confirm if subclause F.8.3.2 is needed to mark only the pictures with the same value of nuh_layer_id.]

– If the current picture is an IRAP picture with NoRaslOutputFlag equal to 1 and with nuh_layer_id equal to 0 that is not picture 0, the following ordered steps are applied:

1. The variable NoOutputOfPriorPicsFlag is derived for the decoder under test as follows:
   - If the current picture is a CRA picture, NoOutputOfPriorPicsFlag is set equal to 1 (regardless of the value of no_output_of_prior_pics_flag).
   - Otherwise, if the value of pic_width_in_luma_samples, pic_height_in_luma_samples, or sps_max_dec_pic_buffering_minus1[HighestTid] derived from the active SPS is different from the value of pic_width_in_luma_samples, pic_height_in_luma_samples, or sps_max_dec_pic_buffering_minus1[HighestTid], respectively, derived from the SPS active for the preceding picture, NoOutputOfPriorPicsFlag may (but should not) be set to 1 by the decoder under test, regardless of the value of no_output_of_prior_pics_flag.

     NOTE 7 – Although setting NoOutputOfPriorPicsFlag equal to no_output_of_prior_pics_flag is preferred under these conditions, the decoder under test is allowed to set NoOutputOfPriorPicsFlag to 1 in this case.

   - Otherwise, NoOutputOfPriorPicsFlag is set equal to no_output_of_prior_pics_flag.
2. The value of NoOutputOfPriorPicsFlag derived for the decoder under test is applied for the HRD as follows:

- If NoOutputOfPriorPicsFlag is equal to 1, all picture storage buffers in the DPB are emptied without output of the pictures they contain, and the DPB fullness is set equal to 0.
- Otherwise (NoOutputOfPriorPicsFlag is equal to 0), all picture storage buffers containing a picture that is marked as "not needed for output" and "unused for reference" are emptied (without output), and all non-empty picture storage buffers in the DPB are emptied by repeatedly invoking the "bumping" process specified in subclause F.13.5.2.4, and the DPB fullness is set equal to 0.

– Otherwise (the current picture is not an IRAP picture with NoRaslOutputFlag equal to 1 or with nuh_layer_id not equal to 0), all picture storage buffers containing a picture which are marked as "not needed for output" and "unused for reference" are emptied (without output). For each picture storage buffer that is emptied, the DPB fullness is decremented by one. The variable currLayerId is set equal to nuh_layer_id of the current decoded picture and when one or more of the following conditions are true, the "bumping" process specified in subclause F.13.5.2.4 is invoked repeatedly while further decrementing the DPB fullness by one for each additional picture storage buffer that is emptied, until none of the following conditions are true:
  - The number of pictures with nuh_layer_id equal to currLayerId in the DPB that are marked as "needed for output" is greater than sps_max_num_reorder_pics[HighestTid] from the active SPS (when currLayerId is equal to 0) or from the active layer SPS for the value of currLayerId (when currLayerId is not equal to 0). [Ed. (CY): to simplify the text for extensions, it is better to use a consistent term of *active SPS* for *"active SPS"* and *"active layer SPS"*. This might require we editorially modify the version 1 spec., specifying only one active SPS is allowed *for nuh_layer_id equal to 0* of a coded video sequence. Similar change may be done for PPS.]
  - sps_max_latency_increase_plus1[HighestTid] of the active SPS (when currLayerId is equal to 0) or the active layer SPS for the value of currLayerId is not equal to 0 and there is at least one picture with nuh_layer_id equal to currLayerId in the DPB that is marked as "needed for output" for which the associated variable PicLatencyCount[currLayerId] is greater than or equal to SpsMaxLatencyPictures[HighestTid] derived from the active SPS (when currLayerId is equal to 0) or from the active layer SPS for the value of currLayerId.
  - The number of pictures with nuh_layer_id equal to currLayerId in the DPB is greater than or equal to sps_max_dec_pic_buffering_minus1[HighestTid] + 1 from the active SPS (when currLayerId is equal to 0) or from the active layer SPS for the value of currLayerId.

F.13.5.2.3 Picture decoding, marking, additional bumping, and storage

The processes specified in this subclause happen instantaneously when the last decoding unit of access unit n containing the current picture is removed from the CPB.

The variable currLayerId is set equal to nuh_layer_id of the current decoded picture.

For each picture in the DPB that is marked as "needed for output" and that has a nuh_layer_id value equal to currLayerId, the associated variable PicLatencyCount[currLayerId] is set equal to PicLatencyCount[currLayerId] + 1.

The current picture is considered as decoded after the last decoding unit of the picture is decoded. The current decoded picture is stored in an empty picture storage buffer in the DPB, and the following applies:

– If the current decoded picture has PicOutputFlag equal to 1, it is marked as "needed for output" and its associated variable PicLatencyCount[currLayerId] is set equal to 0.

– Otherwise (the current decoded picture has PicOutputFlag equal to 0), it is marked as "not needed for output".

The current decoded picture is marked as "used for short-term reference".

When one or more of the following conditions are true, the "bumping" process specified in subclause F.13.5.2.4 is invoked repeatedly until none of the following conditions are true:

– The number of pictures with nuh_layer_id equal to currLayerId in the DPB that are marked as "needed for output" is greater than sps_max_num_reorder_pics[HighestTid] from the active SPS (when currLayerId is equal to 0) or from the active layer SPS for the value of currLayerId, if not equal to 0.

– sps_max_latency_increase_plus1[HighestTid] is not equal to 0 and there is at least one picture with nuh_layer_id equal to currLayerId in the DPB that is marked as "needed for output" for which the associated variable PicLatencyCount[currLayerId] that is greater than or equal to SpsMaxLatencyPictures[HighestTid] derived from the active SPS (when currLayerId is equal to 0) or from the active layer SPS for the value of currLayerId (when currLayerId is not equal to 0).

F.13.5.2.4 "Bumping" process

The "bumping" process consists of the following ordered steps:

1. The pictures that are first for output are selected as the ones having the smallest value of PicOrderCntVal of all pictures in the DPB marked as "needed for output".

2. These pictures are cropped, using the conformance cropping window specified in the active SPS for the picture with nuh_layer_id equal to 0 or in the active layer SPS for a non-zero nuh_layer_id value equal to that of the picture, the cropped pictures are output in ascending order of nuh_layer_id, and the pictures are marked as "not needed for output".
3. Each picture storage buffer that contains a picture marked as "unused for reference" and that included one of the pictures that was cropped and output is emptied.

F.14 SEI messages

The specifications in Annex D together with the extensions and modifications specified in this subclause apply.

[Ed. (CY): to check the semantics in D.3 and that in F.14.2 to make them align with the AU definition.]

F.14.1 SEI message syntax

F.14.1.1 Layer dependency change SEI message syntax

layer_dependency_change(payloadSize) {	**Descriptor**
active_vps_id	u(4)
for(i = 1; i <= vps_max_layers_minus1; i++)	
for(j = 0; j < NumDirectRefLayers[layer_id_in_nuh[i]]; j++)	
ref_layer_disable_flag[i][j]	u(1)
}	

F.14.1.2 Layers present SEI message syntax

layers_present(payloadSize) {	**Descriptor**
lp_sei_active_vps_id	u(4)
for(i = 0; i <= vps_max_layers_minus1; i++)	
layer_present_flag[i]	u(1)
}	

F.14.2 SEI message semantics

Table F-2 – Persistence scope of SEI messages (informative)

SEI message	Persistence scope
Layer dependency change	The access unit containing the SEI message and up to but not including the next access unit, in decoding order, that contains a layers dependency change SEI message or the end of the CVS, whichever is earlier in decoding order
Layers present	The access unit containing the SEI message and up to but not including the next access unit, in decoding order, that contains a layers present change SEI message or the end of the CVS, whichever is earlier in decoding order

F.14.2.1 Layer dependency change SEI message semantics

This SEI message indicates that the layer dependency information changes starting with the current access unit containing the SEI message and is always interpreted with respect to the active VPS. When present, the layer dependency change SEI message applies to the target access unit set that consists of the current access unit and all the subsequent access units, in decoding order, until the next layer dependency change SEI message or the end of the CVS, whichever is earlier in decoding order.

NOTE 8 – The reference layers for any layer are always a subset of those indicated by the active VPS.

NOTE 9 – Layer dependency change SEI messages do not have a cumulative effect.

Some of the layers indicated by the following syntax elements may not be present in the target access unit set.

active_vps_id identifies an active VPS that contains the layer dependency relationship information. The value of active_vps_id shall be equal to the value of video_parameter_set_id of the active VPS for the VCL NAL units of the access unit containing the SEI message.

ref_layer_disable_flag[i][j] equal to 1 indicates that no picture with nuh_layer_id equal to RefLayerId[layer_id_in_nuh[i]][j] is present in any of the reference picture lists after reference picture list modification for pictures with nuh_layer_id equal to layer_id_in_nuh[i] within the target access unit set. ref_layer_disable_flag[i][j] equal to 0 indicates pictures with nuh_layer_id equal to RefLayerId[layer_id_in_nuh[i]][j] may be present in the reference picture lists after reference picture list modification for pictures with nuh_layer_id equal to layer_id_in_nuh[i] within the target access unit set. ref_layer_disable_flag[i][j] shall be equal to 1, if ref_layer_disable_flag[i][j] was equal to 1 in an earlier layer dependency change SEI message for the same CVS. [Ed. (JB): Should double-check semantics, given other inter-layer reference signaling adoptions.]

F.14.2.2 Layers present SEI message semantics

The layers present SEI message provides a mechanism for signalling that NAL units of particular layers indicated by the video parameter set are not present in a particular set of access units. [Ed. (YK): It seems that it is indeed better to call this SEI message as layers not present SEI message.]

When present, the layers present SEI message applies to the access unit containing the SEI message and up to but not including the next access unit, in decoding order, that contains a layers present change SEI message or the end of the CVS, whichever is earlier in decoding order.

A layers present SEI message shall not be included in a scalable nesting SEI message.

A layers present SEI message shall not be included in a NAL unit with TemporalId greater than 0.

lp_sei_active_vps_id identifies an active VPS that contains the information about the layers in the CVS. The value of lp_sei_active_vps_id shall be equal to the value of vps_video_parameter_set_id of the active VPS for the VCL NAL units of the access unit containing the SEI message.

layer_present_flag[i] equal to 1 indicates that there may or may not be NAL units in the target access units with nuh_layer_id equal to layer_id_in_nuh[i]. layer_present_flag[i] equal to 0 indicates that there are no NAL units in the target access units with nuh_layer_id equal to layer_id_in_nuh[i]. [Ed. (YK): The definition of "the target access units" is missing.]

When layer_present_flag[i] is equal to 1 and i is greater than 0, layer_present_flag[LayerIdxInVps[RefLayerId[layer_id_in_nuh[i]][j]]] shall be equal to 1 for all values of j in the range of 0 to NumDirectRefLayers[layer_id_in_nuh[i]] − 1, inclusive.

F.15 Video usability information

F.15.1 General

The specifications in Annex E.1 apply.

F.15.2 VUI syntax

The specifications in Annex E.2 apply.

F.15.2.1 VUI parameters syntax

vui_parameters( ) {	**Descriptor**
aspect_ratio_info_present_flag	u(1)
if(aspect_ratio_info_present_flag) {	
aspect_ratio_idc	u(8)
if(aspect_ratio_idc = = EXTENDED_SAR) {	
sar_width	u(16)
sar_height	u(16)
}	
}	
overscan_info_present_flag	u(1)
if(overscan_info_present_flag)	
overscan_appropriate_flag	u(1)
video_signal_type_present_flag	u(1)
if(video_signal_type_present_flag) {	
video_format	u(3)
video_full_range_flag	u(1)
colour_description_present_flag	u(1)
if(colour_description_present_flag) {	
colour_primaries	u(8)
transfer_characteristics	u(8)
matrix_coeffs	u(8)
}	
}	
chroma_loc_info_present_flag	u(1)
if(chroma_loc_info_present_flag) {	
chroma_sample_loc_type_top_field	ue(v)
chroma_sample_loc_type_bottom_field	ue(v)
}	
neutral_chroma_indication_flag	u(1)
field_seq_flag	u(1)
frame_field_info_present_flag	u(1)
default_display_window_flag	u(1)
if(default_display_window_flag) {	
def_disp_win_left_offset	ue(v)
def_disp_win_right_offset	ue(v)
def_disp_win_top_offset	ue(v)
def_disp_win_bottom_offset	ue(v)
}	
vui_timing_info_present_flag	u(1)
if(vui_timing_info_present_flag) {	
vui_num_units_in_tick	u(32)
vui_time_scale	u(32)
vui_poc_proportional_to_timing_flag	u(1)
if(vui_poc_proportional_to_timing_flag)	
vui_num_ticks_poc_diff_one_minus1	ue(v)
vui_hrd_parameters_present_flag	u(1)

if(vui_hrd_parameters_present_flag)	
hrd_parameters(1, sps_max_sub_layers_minus1)	
}	
bitstream_restriction_flag	u(1)
if(bitstream_restriction_flag) {	
tiles_fixed_structure_flag	u(1)
if (nuh_layer_id > 0)	
tile_boundaries_aligned_flag	u(1)
motion_vectors_over_pic_boundaries_flag	u(1)
restricted_ref_pic_lists_flag	u(1)
min_spatial_segmentation_idc	ue(v)
max_bytes_per_pic_denom	ue(v)
max_bits_per_min_cu_denom	ue(v)
log2_max_mv_length_horizontal	ue(v)
log2_max_mv_length_vertical	ue(v)
}	
}	

F.15.2.1.1 Sequence parameter set extension VUI parameters syntax

sps_extension_vui_parameters( ) {	**Descriptor**
if(bitstream_restriction_flag) {	
num_ilp_restricted_ref_layers	ue(v)
for(i = 0; i < num_ilp_restricted_ref_layers; i++) {	
min_spatial_segment_offset_plus1[i]	ue(v)
if(min_spatial_segment_offset[i] > 0) {	
ctu_based_offset_enabled_flag[i]	u(1)
if(ctu_based_offset_enabled_flag[i])	
min_horizontal_ctu_offset_plus1[i]	ue(v)
}	
}	
}	
}	

F.15.2.2 HRD parameters syntax

The specifications in Annex E.2.2 apply.

F.15.2.3 Sub-layer HRD parameters syntax

The specifications in Annex E.2.3 apply.

F.15.3 VUI semantics

F.15.3.1 VUI parameters semantics

The specifications in Annex E.3.1 apply with the following changes.

tile_boundaries_aligned_flag equal to 1 indicates that, when any two samples of one picture in an access unit belong to one tile, the collocated samples, if any, in another picture in the same access unit belong to one tile, and when any two samples of one picture in an access unit belong to different tiles, the collocated samples in another picture in the same

access unit shall belong to different tiles. tile_boundaries_aligned_flag equal to 0 indicates that such a restriction may or may not apply.

[Ed. (YK): Define "collocated sample".]

[Ed. (GT): Remarks by Jill: Do we want to impose the restriction for ALL pictures in the access unit, or only for a current picture and its inter-layer reference picture(s)? Semantic restriction could be placed on the tile parameters when this VUI flag is set.]

F.15.3.1.1 Sequence parameter set extension VUI parameters semantics

num_ilp_restricted_ref_layers equal to 0 indicates that no additional restriction on inter-layer prediction applies for any of the direct reference layers of any layer referring to the SPS. num_ilp_restricted_ref_layers greater than 0 specifies, for any layer referring to the SPS, the number of direct reference layers from which inter-layer prediction is additionally restricted as specified below.

When num_ilp_restricted_ref_layers is greater than 0, it is a requirement of bitstream conformance that all the following conditions are true:

- The value of num_ilp_restricted_ref_layers shall be equal to NumDirectRefLayer[nuh_layer_id], where nuh_layer_id is the nuh_layer_id of any picture referring to the SPS.
- All layers referring to the SPS shall have the same number of direct reference layers as specified by the active VPS.

For any layer SPS RBSP being activated for a non-zero nuh_layer_id equal to CurrLayerId when a picture is decoded, the i-th direct reference layer has nuh_layer_id equal to RefLayerId[currLayerId][i]. [Ed. (JB): Language should be checked and futher improved.]

The variables refCtbLog2SizeY[i], refPicWidthInCtbsY[i], and refPicHeightInCtbsY[i] are set equal to CtbLog2SizeY, PicWidthInCtbsY, and PicHeightInCtbsY, respectively, of the i-th direct reference layer.

[Ed. (YK): Consider using better syntax element names for min_spatial_segment_offset_plus1[i], ctu_based_offset_enabled_flag[i], and min_horizontal_ctu_offset_plus1[i].]

min_spatial_segment_offset_plus1[i] indicates the spatial area, in each picture of the i-th direct reference layer, that is not used for inter-layer prediction for decoding of a picture referring to the SPS, by itself or together with min_horizontal_ctu_offset_plus1[i], as specified below. The value of min_spatial_segment_offset_plus1[i] shall be in the range of 0 to refPicWidthInCtbsY[i] * refPicHeightInCtbsY[i], inclusive. When not present, the value of min_spatial_segment_offset_plus1[i] is inferred to be equal to 0.

ctu_based_offset_enabled_flag[i] equal to 1 specifies that the spatial area, in units of CTUs, in each picture of the i-th direct reference layer, that is not used for inter-layer prediction for decoding of a picture referring to the SPS is indicated by min_spatial_segment_offset_plus1[i] and min_horizontal_ctu_offset_plus1[i] together. ctu_based_offset_enabled_flag[i] equal to 0 specifies that the spatial area, in units of slice segments, tiles, or CTU rows, in each picture of the i-th direct reference layer, that is not used for inter-layer prediction for decoding of a picture referring to the SPS is indicated by min_spatial_segment_offset_plus1[i] only. When not present, the value of ctu_based_offset_enabled_flag[i] is inferred to be equal to 0.

min_horizontal_ctu_offset_plus1[i], when ctu_based_offset_enabled_flag[i] is equal to 1, indicates the spatial area, in each picture of the i-th direct reference layer, that is not used for inter-layer prediction for decoding of a picture referring to the SPS, together with min_spatial_segment_offset_plus1[i], as specified below. The value of min_horizontal_ctu_offset_plus1[i] shall be in the range of 0 to refPicWidthInCtbsY[i], inclusive.

When ctu_based_offset_enabled_flag[i] is equal to 1, the variable minHorizontalCtbOffset[i] is derived as follows:

$$\begin{aligned} \text{minHorizontalCtbOffset}[i] = {} & (\text{min\_horizontal\_ctu\_offset\_plus1}[i] > 0)\ ? \\ & (\text{min\_horizontal\_ctu\_offset\_plus1}[i] - 1) : (\text{refPicWidthInCtbsY}[i] - 1) \end{aligned} \tag{F-14}$$

The variable colCtbAddr[i] that denotes the raster scan address of the collocated CTU, in a picture in the i-th direct reference layer, of the CTU with raster scan address equal to ctbAddr in a picture referring to the SPS is derived as follows [Ed. (YK): Define "collocated CTU".]:

$$\text{xAddrOfCtb}[i] = (\text{ctbAddr} \,\%\, \text{PicWidthInCtbsY}) \ll \text{CtbLog2SizeY} \tag{F-15}$$

$$\text{yAddrOfCtb}[i] = (\text{ctbAddr} / \text{PicWidthInCtbsY}) \ll \text{CtbLog2SizeY} \tag{F-16}$$

$$\text{xColCtb}[i] = \text{xAddrOfCtb}[i] \gg \text{refCtbLog2SizeY}[i] \tag{F-17}$$

$$\text{yColCtb}[i] = \text{yAddrOfCtb}[i] \gg \text{refCtbLog2SizeY}[i] \tag{F-18}$$

colCtbAddr[i] = xColCtb[i] + (yColCtb[i] * refPicWidthInCtbsY[i]) (F-19)

When min_spatial_segment_offset_plus1[i] is greater than 0, it is a requirement of bitstream conformance that the following shall apply:

- If ctu_based_offset_enabled[i] is equal to 0, exactly one of the following shall apply:
  - In each PPS referred to by a picture in the i-th direct reference layer, tiles_enabled_flag is equal to 0 and entropy_coding_sync_enabled_flag is equal to 0, and the following applies:
    - Let slice segment A be any slice segment referring to the SPS and ctbAddr be the raster scan address of the last CTU in slice segment A. Let slice segment B be the slice segment that belongs to the same access unit as slice segment A, belongs to the i-th direct reference layer, and contains the CTU with raster scan address colCtbAddr[i]. Let slice segment C be the slice segment that is in the same picture as slice segment B and follows slice segment B in decoding order, and between slice segment B and that slice segment there are min_spatial_segment_offset_plus1[i] - 1 slice segments in decoding order. When slice segment C is present, the syntax elements of slice segment A are constrained such that no sample or syntax elements values in slice segment C or any slice segment of the same picture following C in bitstream order are used for inter-layer prediction in the decoding process of any samples within slice segment A.
  - In each PPS referred to by a picture in the i-th direct reference layer, tiles_enabled_flag is equal to 1 and entropy_coding_sync_enabled_flag is equal to 0, and the following applies:
    - Let tile A be any tile in any picture picA referring to the SPS and ctbAddr be the raster scan address of the last CTU in tile A. Let tile B be the tile that is in the picture picB belonging to the same access unit as picA and belonging to the i-th direct reference layer and that contains the CTU with raster scan address colCtbAddr[i]. Let tile C be the tile that is also in picB and follows tile B in decoding order, and between tile B and that tile there are min_spatial_segment_offset_plus1[i] - 1 tiles in decoding order. When slice segment C is present, the syntax elements of tile A are constrained such that no sample or syntax elements values in tile C or any tile of the same picture following C in bitstream order are used for inter-layer prediction in the decoding process of any samples within tile A.
  - In each PPS referred to by a picture in the i-th direct reference layer, tiles_enabled_flag is equal to 0 and entropy_coding_sync_enabled_flag is equal to 1, and the following applies:
    - Let CTU row A be any CTU row in any picture picA referring to the SPS and ctbAddr be the raster scan address of the last CTU in CTU row A. Let CTU row B be the CTU row that is in the picture picB belonging to the same access unit as picA and belonging to the i-th direct reference layer and that contains the CTU with raster scan address colCtbAddr[i]. Let CTU row C be the CTU row that is also in picB and follows CTU row B in decoding order, and between CTU row B and that CTU row there are min_spatial_segment_offset_plus1[i] - 1 CTU rows in decoding order. When CTU row C is present, the syntax elements of CTU row A are constrained such that no sample or syntax elements values in CTU row C or row of the same picture following C are used for inter-layer prediction in the decoding process of any samples within CTU row A.
- Otherwise (ctu_based_offset_enabled_flag[i] is equal to 1), the following applies:
  - The variable refCtbAddr[i] is derived as follows:

    xOffset[i] = ((xColCtb[i] + minHorizontalCtbOffset[i]) > (refPicWidthInCtbsY[i] - 1)) ? (refPicWidthInCtbsY[i] - 1 –xColCtb[i]) : (minHorizontalCtbOffset[i]) (F-20)

    yOffset[i] = (min_spatial_segment_offset_plus1[i] – 1) * refPicWidthInCtbsY[i] (F-21)

    refCtbAddr[i] = colCtbAddr[i] + xOffset[i] + yOffset[i] (F-22)

  - Let CTU A be any CTU in any picture picA referring to the SPS, and ctbAddr be the raster scan address ctbAddr of CTU A. Let CTU B be a CTU that is in the picture belonging to the same access unit as picA and belonging to the i-th direct reference layer and that has raster scan address greater than refCtbAddr[i]. When CTU B is present, the syntax elements of CTU A are constrained such that no sample or syntax elements values in CTU B are used for inter-layer prediction in the decoding process of any samples within CTU A.

F.15.3.2 HRD parameters semantics

The specifications in Annex E.3.2 apply.

F.15.3.3 Sub-layer HRD parameters semantics

The specifications in Annex E.3.3 apply.

Annex G

Multiview coding

(This annex forms an integral part of this Recommendation | International Standard)

This annex specifies multiview coding that use the syntax, semantics, and decoding processes specified in clauses 2-9 and Annex A-F.

G.1 Scope

Decoding processes and bitstreams conforming to this annex are completely specified in this annex with reference made to clauses 2-9 and Annexes A-F.

G.2 Normative references

The specifications in clause 2 apply.

G.3 Definitions

The specification in clause F.3 apply.

G.4 Abbreviations

The specification in clause 4 apply.

G.5 Conventions

The specification in clause 5 apply.

G.6 Source, coded, decoded and output data formats, scanning processes, and neighbouring relationships

The specification in clause 6 apply.

G.7 Syntax and semantics

The specification in clause F.7 apply.

G.8 Decoding processes

G.8.1 General decoding process

The specification in subclause F.8.1 applies.

G.8.1.1 Decoding process for a coded picture with nuh_layer_id greater than 0

The decoding process operates as follows for the current picture CurrPic:

1. The decoding of NAL units is specified in subclause G.8.2.
2. The processes in subclause G.8.1.2 and G.8.3.4 specify the following decoding processes using syntax elements in the slice segment layer and above:
   - Prior to decoding the first slice of the current picture, subclause G.8.1.2 is invoked.
   - At the beginning of the decoding process for each P or B slice, the decoding process for reference picture lists construction specified in subclause G.8.3.4 is invoked for derivation of reference picture list 0 (RefPicList0), and when decoding a B slice, reference picture list 1 (RefPicList1).
3. The processes in subclauses G.8.4, G.8.5, G.8.6, and G.8.7 specify decoding processes using syntax elements in all syntax structure layers. It is a requirement of bitstream conformance that the coded slices of the picture shall contain slice segment data for every coding tree unit of the picture, such that the division of the picture into

slices, the division of the slices into slice segments, and the division of the slice segments into coding tree units each form a partitioning of the picture.

4. After all slices of the current picture have been decoded, the marking process for ending the decoding of a coded picture with nuh_layer_id greater than 0 specified in subclause G.8.1.3 is invoked.

G.8.1.2 Decoding process for inter-layer reference picture set

Output of this process is an updated list of inter-layer pictures RefPicSetInterLayer.

The list RefPicSetInterLayer is first emptied and then derived as follows.

```
for( i = 0; i < NumActiveRefLayerPics; i++ ) {
    if( there is a picture picX in the DPB that is in the same access unit as the current picture and has
            nuh_layer_id equal to RefPicLayerId[ i ] ) {
        RefPicSetInterLayer[ i ] = picX
        RefPicSetInterLayer[ i ] is marked as "used for long-term reference"
    } else
        RefPicSetInterLayer[ i ] = "no reference picture"
}
```

There shall be no entry equal to "no reference picture" in RefPicSetInterLayer.

If the current picture is a RADL picture, there shall be no entry in the RefPicSetInterLayer that is a RASL picture.

NOTE 10 – An access unit may contain both RASL and RADL pictures.

G.8.1.3 Marking process for ending the decoding of a coded picture with nuh_layer_id greater than 0

Output of this process is:

– a potentially updated marking as "used for short-term reference" for some decoded pictures.

The following applies.

```
for( i = 0; i <  NumActiveRefLayerPics; i++ )
    RefPicSetInterLayer[ i ] is marked as "used for short-term reference"
```

G.8.2 NAL unit decoding process

The specification in subclause 8.2 apply.

G.8.3 Slice decoding processes

G.8.3.1 Decoding process for picture order count

The specifications in subclause 8.3.1 apply.

G.8.3.2 Decoding process for reference picture set

The specifications in subclause 8.3.2 apply.

G.8.3.3 Decoding process for generating unavailable reference pictures

The specifications in subclause 8.3.3 apply.

G.8.3.4 Decoding process for reference picture lists construction

The specifications in subclause F.8.3.4 apply.

G.8.4 Decoding process for coding units coded in intra prediction mode

The specifications in subclause 8.4 apply.

G.8.5 Decoding process for coding units coded in inter prediction mode

The specifications in subclause F.8.5.apply.

G.8.6 Scaling, transformation and array construction process prior to deblocking filter process

The specifications in subclause 8.6 apply.

G.8.7 In-loop filter process

The specifications in subclause 8.7 apply.

G.9 Parsing process

The specifications in clause 9 apply.

G.10 Specification of bitstream subsets

The specifications in clause 10 apply.

G.11 Profiles, tiers, and levels

G.11.1 Profiles

G.11.1.1 General

TBD.

G.11.1.2 Stereo Main profile

Bitstreams conforming to the Stereo Main profile shall obey the following constraints:

– The sub-bitstream resulting from the sub-bitstream extraction process with any value of tIdTarget and a value of 0 in layerIdListTarget as inputs shall conform to the Main profile.

– The bitstream shall contain one layer with nuh_layer_id equal to i for which ViewScalExtLayerFlag[i] is equal to 1.

– When ViewScalExtLayerFlag[i] is equal to 1, inter_view_mv_vert_constraint_flag shall be equal to 1 in the sps_extension( ) syntax structure of the active layer SPS RBSP of any coded pictures with nuh_layer_id equal to i.

– When ViewScalExtLayerFlag[i] is equal to 1, ScalabilityId[LayerIdxInVps [i]][smIdx] shall be equal to 0 for smIdx in the range of 1 to 15, inclusive, for any coded picture with nuh_layer_id equal to i. SPSs shall have scaled_ref_layer_offset_present_flag equal to 0 only.

[Ed. (GT) Consider following text as suggested by Miska:

– The bitstream shall contain a sub-bitstream consisting of two layers having nuh_layer_id equal to 0 and nuhLayerIdA for which ScalabilityId[LayerIdxInVps [nuhLayerIdA]][smIdx] shall be equal to 0 for smIdx in the range of 1 to 15 , inclusive.

]

– [Ed. (JB): consider adding the following constraint: avc_base_layer_flag shall be equal to 0.]

G.11.2 Tiers and levels

TBD

G.12 Byte stream format

The specifications in subclause F.12 apply.

G.13 Hypothetical reference decoder

The specifications in subclause F.13 and its subclauses apply.

G.14 SEI messages

The specifications in Annex D and subclause F.14 together with the extensions and modifications specified in this subclause apply.

G.14.1 SEI message syntax

G.14.1.1 3D reference displays information SEI message syntax

three_dimensional_reference_displays_info(payloadSize) {	**Descriptor**
prec_ref_display_width	ue(v)
ref_viewing_distance_flag	u(1)
if(ref_viewing_distance_flag)	
prec_ref_viewing_dist	ue(v)
num_ref_displays_minus1	ue(v)
for(i = 0; i <= num_ref_displays_minus1; i++) {	
left_view_id[i]	ue(v)
right_view_id[i]	ue(v)
exponent_ref_display_width[i]	u(6)
mantissa_ref_display_width[i]	u(v)
if(ref_viewing_distance_flag) {	
exponent_ref_viewing_distance[i]	u(6)
mantissa_ref_viewing_distance[i]	u(v)
}	
additional_shift_present_flag[i]	u(1)
if (additional_shift_present[i])	
num_sample_shift_plus512[i]	u(10)
}	
three_dimensional_reference_displays_extension_flag	u(1)
}	

G.14.2 SEI message semantics

Table G-3 – Persistence scope of SEI messages (informative)

SEI message	Persistence scope
3D reference displays information	[Ed. (GT): To be clarified. See note below.]

G.14.2.1 3D reference displays information SEI message semantics

A reference displays information message contains information about the reference display width(s) and reference viewing distance(s) as well as information about the corresponding baseline distance(s) and additional horizontal image shift(s), which form a stereo-pair for the reference display width and the reference viewing distance. This information enables a view renderer to produce a proper stereo-pair for the target screen width and the viewing distance. The reference display width and viewing distance values are signalled in units of centimetres. The reference baseline distance may be inferred as the reference baseline between the left and the right views in a reference pair.

When present, this SEI message shall be associated with an IRAP access unit. The baseline and shift information signalled for the reference display is valid for all access units they associated with and until the next IRAP access unit or the next access unit containing reference displays information SEI message. [Ed. (GT): Persistence scope needs to be clarified, particularly whether coding order or display order applies.]

NOTE 11 – The reference displays information SEI message specifies display parameters for which the 3D sequence was optimized and the corresponding reference parameters. Each reference display (i.e. a reference display width and possibly a corresponding viewing distance) is associated with one reference pair of views by signalling their ViewId.

The following equations can be used for calculating the baseline distance and horizontal shift for the receiver's display when the ratio between the receiver's viewing distance and the reference viewing distance is the same as the ratio between the receiver screen width and the reference screen width:

baseline = refBaseline * (refDisplayWidth ÷ displayWidth)

shift = refShift * (refDisplayWidth ÷ displayWidth)

In case the reference baseline and shift are not present in the bitstream and the SEI messages, other view synthesis parameters dependent on the baseline and the shift should be scaled instead using similar equations.

parameter = refParameter * (refDisplayWidth ÷ displayWidth)

The reference baseline, the reference shift or other view synthesis parameters are the baseline and the shift or other view synthesis parameters between the two views corresponding to the left_view_id and the right_view_id. The reference baseline and shift (or other view synthesis parameters) can be extracted from the bitstream or SEI messages. In the provided equations, the width of the visible part of the display used for showing the video sequence should be understood under "display width". The same equations

can also be used for choosing the baseline distance and horizontal shift or other view synthesis parameters in cases when the viewing distance is not scaled proportionally to the screen width compared to the reference display parameters. In this case, the effect of applying these equations would be to keep the perceived depth in the same proportion to the viewing distance as in the reference setup.

When baseline or view synthesis parameters are updated by an SEI message or in the bitstream in a following access unit and the baseline or other view synthesis parameters between the views corresponding to the left_view_id and the right_view_id in the following access unit change relative to those in the access unit which the reference displays information SEI belongs to, the baseline and the horizontal shift (or other view synthesis parameters) for the receiver's display in the following access unit should be modified accordingly. That means that the ratio of the baseline (or another view synthesis parameter) between the two views used in the view synthesis and the baseline (or another view synthesis parameter) between view corresponding to the left_view_id and the right_view_id in the following access unit should be equal to the ratio of the same parameters between two views used in the view synthesis and the parameters between the view corresponding to the left_view_id and the right_view_id in the access unit, which the reference displays information SEI message belongs to The horizontal shift for the receiver's display should also be modified by scaling it with the same factor as that used to scale the baseline distance (or other view synthesis parameters).

[Ed. (GT): Above note is about parameters that are not included in current MV-HEVC SEI messages, therefore it should be considered to remove it.]

prec_ref_display_width specifies the exponent of the maximum allowable truncation error for refDisplayWidth[i] as given by $2^{-\text{prec\_ref\_display\_width}}$. The value of prec_ref_display_width shall be in the range of 0 to 31, inclusive.

ref_viewing_distance_flag equal to 1 indicates the presence of reference viewing distance. ref_viewing_distance_flag equal to 0 indicates that the reference viewing distance is not present in the bitstream.

prec_ref_viewing_dist specifies the exponent of the maximum allowable truncation error for refViewingDist[i] as given by $2^{-\text{prec\_ref\_viewing\_dist}}$. The value of prec_ref_viewing_dist shall be in the range of 0 to 31, inclusive.

num_ref_displays_minus1 plus 1 specifies the number of reference displays that are signalled in the bitstream. The value of num_ref_displays_minus1 shall be in the range of 0 to 31, inclusive.

left_ref_view_id[i] specifies the ViewId of the left view of a stereo-pair corresponding to the i-th reference display.

right_ref_view_id[i] specifies the ViewId of the right view of a stereo-pair corresponding to the i-th reference display.

exponent_ref_display_width[i] specifies the exponent part of the reference display width of the i-th reference display. The value of exponent_ref_display_width[i] shall be in the range of 0 to 62, inclusive. The value 63 is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified reference display width.

mantissa_ref_display_width[i] specifies the mantissa part of the reference display width of the i-th reference display. The variable refDispWidthBits specifying the number of bits of the mantissa_ref_display_width[i] syntax element is derived as follows:

– If exponent_ref_display_width[i] is equal to 0, refDispWidthBits is set equal to Max(0, prec_ref_display_width − 30).

– Otherwise (0 < exponent_ref_display_width[i] < 63), refDispWidthBits is set equal to Max(0, exponent_ref_display_width[i] + prec_ref_display_width − 31).

exponent_ref_viewing_distance[i] specifies the exponent part of the reference viewing distance of the i-th reference display. The value of exponent_ref_viewing_distance[i] shall be in the range of 0 to 62, inclusive. The value 63 is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified reference display width.

mantissa_ref_viewing_distance[i] specifies the mantissa part of the reference viewing distance of the i-th reference display. The variable refViewDistBits specifying the number of bits of the mantissa_ref_viewing_distance[i] syntax element is derived as follows;

– If exponent_ref_viewing_distance[i] is equal to 0, the refViewDistBits is set equal to Max(0, prec_ref_viewing_distance − 30).

– Otherwise (0 < exponent_ref_viewing_distance[i] < 63), refViewDistBits is set equal to Max(0, exponent_ref_viewing_distance[i] + prec_ref_viewing_distance − 31).

The variables in the x row of Table G-4 are derived as follows from the respective variables or values in the e, n, and v rows of Table G-4 as follows.

– If e is not equal to 0, the following applies:

$$x = 2^{(e-31)} * (1 + n \div 2^{v}) \quad \text{(G-2)}$$

– Otherwise (e is equal to 0), the following applies:

$$x = 2^{-(30+v)} * n \quad \text{(G-3)}$$

NOTE 12 – The above specification is similar to that found in IEC 60559 60559:1989, *Binary floating-point arithmetic for microprocessor systems*.

Table G-4 – Association between camera parameter variables and syntax elements

x	refDisplayWidth[i]	refViewingDistance[i **]**
e	exponent_ref_display_width[i]	exponent_ref_viewing_distance[i]
n	mantissa_ref_display_width[i]	mantissa_ref_viewing_distance[i]
v	refDispWidthBits	refViewDistBits

additional_shift_present_flag[i] equal to 1 indicates that the information about additional horizontal shift of the left and right views for the i-th reference display is present in the bitstream. additional_shift_present_flag[i] equal to 0 indicates that the information about additional horizontal shift of the left and right views for the i-th reference display is not present in the bitstream.

num_sample_shift_plus512[i] indicates the recommended additional horizontal shift for a stereo-pair corresponding to the i-th reference baseline and the i-th reference display.

– If (num_sample_shift_plus512[i] − 512) is less than 0, it is recommended that the left view of the stereo-pair corresponding to the i-th reference baseline and the i-th reference display is shifted in the left direction by (512 − num_sample_shift_plus512[i]) samples with respect to the right view of the stereo-pair.

– Otherwise, if num_sample_shift_plus512[i] is equal to 512, it is recommended that shifting is not applied.

– Otherwise, ((num_sample_shift_plus512[i] − 512) is greater than 0), it is recommended that the left view in the stereo-pair corresponding to the i-th reference baseline and the i-th reference display should be shifted in the right direction by (512 − num_sample_shift_plus512[i]) samples with respect to the right view of the stereo-pair.

The value of num_sample_shift_plus512[i] shall be in the range of 0 to 1023, inclusive.

three_dimensional_reference_displays_extension_flag equal to 0 indicates that no additional data follows within the reference displays SEI message. The value of three_dimensional_reference_displays_extension_flag shall be equal to 0 in bitstreams conforming to this version of this Specification. The value of 1 for three_dimensional_reference_displays_extension_flag is reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore all data that follows the value of 1 for three_dimensional_reference_displays_extension_flag in a reference displays SEI message.

NOTE 13 – Shifting the left view in the left (or right) direction by x samples with respect to the right view can be performed by the following two-step processing:

1) Shift the left view by x/2 samples in the left (or right) direction, and shift the right view by x/2 samples in the right (or left) direction.

2) Fill the left and right image margins of x/2 samples in width in both the left and right views in background colour.

The following pseudo code explains the recommended shifting processing in the case of shifting the left view in the left direction by x samples with respect to the right view.

```
for( i = x/2; i < width − x/2; i++ )
    for( j=0;  j < height; j++ ) {
        leftView[ j ][ i ] = leftView[ j ][ i + x/2 ]
        rightView[ j ][ width − 1 − i ] = rightView[ j ][ width − 1 − i − x/2 ]
       }
for( i = 0; i < x/2; i++)
    for( j = 0; j < height; j++ ) {
        leftView[ j ][ width − 1 − i ] = leftView[ j ][ i ] = backgroundColour
        rightView[ j ][ width − 1 − i ] = rightView[ j ][ i ] = backgroundColour
  }
```

The following pseudo code explains the recommended shifting processing in the case of shifting the left view in the right direction by x samples with respect to the right view.

```
for( i = x/2; i < width − x/2; i++ )
    for( j = 0; j < height; j++ ){
        leftView[ j ][ width − 1 − i ] = leftView[ j ][ width − 1 − i − x/2 ]
        rightView[ j ][ i ] = rightView[ j ][ i + x/2 ]
```

```
    }
for( i=0; i < x/2; i++ )
     for( j = 0; j < height; j++ ) {
          leftView[ j ][ width − 1 − i ] = leftView[ j ][ i ] = backgroundColour
          rightView[ j ][ width − 1 − i ] = rightView[ j ][ i ] = backgroundColour
    }
```

The variable backgroundColour may take different values in different systems, for example black or gray.

G.15 Video usability information

The specifications in Annex F.15 apply.