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
6th Meeting: Geneva, CH, 25 Oct. – 1 Nov. 2013

Document: JCT3V-F1004-v6

Title:	**MV-HEVC Draft Text 6**		
Status:	Output Document of JCT-3V		
Purpose:	Draft of MV-HEVC		
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

Ed.Notes (Draft 6)

- --------- Release v6 -----------
- Accepted all change marks.
- --------- Release v5 -----------
- (Review GT05)
- (Review GT04)
- (Review YK01)
- --------- Release v4 -----------
- (Review MH05)
- (Review YK00)
- (Review MH04)
- (HRD/O0164/Multilayer HRD) #15 Decision: Adopt, modified as follows: It was suggested to constrain the stalling based on the relative cpb removal times, which must be in decoding order. The "du_based_bpb_sync_flag" is not needed, in view of this. SEI in the highest layer of the layer set or (inclusive "or") VPS VUI is used to carry the parameters (at encoder discretion). SEI in higher layer and SEI in VUI do not need to repeat information available in some lower layer. Shall be after APS SEI and buffering period SEI and before all other SEI of all layers except other HRD related SEI.
- Merged the specifications of DPB operations in subclause F.13 and its subclauses to Annex C and its subclauses.
- Unification of active layer SPS and active SPS.
- (HRD/O0217/Sub-DPB based DPB operations) #13 Decision: Adopt – Specify a separate DPB capacity for each layer – no sharing of capacity across layers – each layer has its own parameters (max pictures, max latency, max reordering). This proposal would specify distinct parameters for each "output layer set" and to change the definition of an operation point to be specific to an output layer set instead of a 'layer set". Decision: Adopted this aspect as well.
- (HRD/O0266/Flushing decoded picture) #14 Decision: Adopt (harmonize with O0149 proposal 3 and supply text in a revision of O0266).
- (Fix MV-HEVC trac 47) Missing a close-bracket on slice_pic_order_cnt_lsb
- (Fix MV-HEVC trac 48) Wrong element name of poc_lsb_present_flag[] on slice_segment_header( ).
- --------- Release v3 -----------
- (Review MH03)
    - Modified the integration of "(RALS/O0139/Prop4) #8 layer initialization picture (LIP)" to support also layer-wise start-up where not all pictures are present in the initial IRAP access unit.
    - Modified the integration of "(POC/O0117/Modify PicOrderCntVal of prevTid0Pic) #35" to differentiate between PrevPicOrderCnt values of different layers
    - Added editor's notes related to "(POC/O0211/Fix ambiguity) #38" on contraints that the approach imposes.
- (POC/O0140,O0213/Ed. Note) #39 Decision (Non-Normative): Add a note to explain what an encoder needs to do to avoid the problem – MMH to provide the wording.
- (RALS/O0139/Prop4) #8 layer initialization picture (LIP): A picture that is an IRAP picture with NoRaslOutputFlag equal to 1 or that is contained in an initial IRAP access unit, of which LayerInitializedFlag[refLayerId] is equal to 1 for all values of refLayerId equal to RefLayerId[nuh_layer_id][j], where j is in the range of 0 to NumDirectRefLayers[nuh_layer_id] − 1, inclusive. Decision (Ed.): Agreed in spirit. Editors to determine exact phrasing.
- (RALS/O0139/Prop2/SPS activation) #7 Decision (Ed): Agreed in spirit that we should not allow activation of a new SPS by an enhancement layer non-IRAP picture that is not the first picture in the bitstream in that enhancement layer (that is not an LIP picture) and should not allow a "normal" CRA in an enhancement layer to activate a different SPS than what was already referred to by the preceding pictures in decoding order in that enhancement layer. (Editors to figure out how to phrase this in specification language.)
- (POC/O0117/Modify PicOrderCntVal of prevTid0Pic) #35 Modification of the PicOrderCntVal of prevTid0Pic and modification to the decoding process for reference picture set, to address problems found for cross-layer POC alignment.
- (POC/O0211/Fix ambiguity) #38 Modify POC derivation to correct an ambiguity in the spec.
- --------- Release v2 -----------
- (Review GT03) Removal of clarified editor's comments.
- (Review MH02): review of scaled reference layer offsets, (SHVC/O0098/Scaled ref layer offset) #36.
- (SHVC/O0098/Scaled ref layer offset) #36 Modify signalling of scaled reference layer offsets to allow signalling of any lower layer, rather than just a direct reference layer, in order to enable alignment of auxiliary pictures. In further JCT-VC and JCT-3V discussion, it was also agreed to use the same offset signalling for MV-HEVC as well as SHVC

- (Review JB02) Move location of chroma_and_bit_depth_vps_present_flag
- (PS/O0179/Rep. Format) #18 Add flag in rep_format( ) syntax structure to control sending of chroma and bit depth related parameters, as proposed in the v2 version of JCTVC-O0179.
- (Review GT02) Minor cleanups, mainly related to F0169.
- (SEI/F0169/depth rep info SEI) #40 Depth representation information SEI message for auxiliary pictures.
- (AUX/O0358/Reserved range) #16 Decision: Define a range of values of auxiliary picture types, the values 0x80-0x8F, for which the interpretation is specified externally or by other information in the bitstream (e.g., some SEI message to be defined later).
- (AUX/O0135/default_one_target_output_flag) #2 Carriage of auxiliary pictures. Decision: Relating to section 6, regarding auxiliary picture ID as part of the definition of the semantics of default_one_target_output_flag, adopt first variant.
- (AUX/O0041/HLS auxiliary picture layers) #1 Decision: Use nuh_layer_id to identify auxiliary pictures and map them to an interpretation (roughly per O0041, as clarified below). Do not make a blanket constraint that prohibits dependencies for auxiliary picture, but impose that constraint for the specific ones listed in O0041 Decision: Adopted the general structure and alpha and depth types. It was agreed that the terminology should be rephrased to not directly link the concepts auxiliary/primary to the concepts of normative/supplemental.
- --------- Release v1 -----------
- (Review JB01)
    - O0142: Added restriction on sps_extension_type_flag[i] in stereo main profile
    - O0096: Modified restriction on number of rep formats to apply to VPS syntax element, not SPS
    - O0120: Renamed sub_layers_vps_max_minus1_present_flag to vps_sub_layers_max_minus1_present_flag
    - O0062: Changed poc_lsb_present_flag to poc_lsb_not_present_flag.
    - O0096: Introduced variables VpsInterLayerSamplePredictionEnabled[i][j] and VpsInterLayerMotionPredictionEnabled[i][j]
- (Review MH01):
    - Typo: the first syntax element in vps_vui( ) is cross_layer_pic_type_aligned_flag in the syntax table.
    - The integration of JCTVC-O0220/Prop2 fixed to refer to the correct decoding process for generating unavailable reference pictures.
    - The integration of JCTVC-O0149/Prop1 moved from F.8.3.2 to 8.3.2.
    - Subclause 8.3.2 corrected to not cause marking of all pictures (with any nuh_layer_id) as "unused for reference" when the base layer contains an IDR picture with cross_layer_bla_flag equal to 0.
- (Review GT01)
- (Review YY01)
- (PS/O0096/rep format syntax element length) #20 Modification of length to 8 bit as decided later in trac.
- (Gen/O0153/output highest layer) #28 Add a flag in the VPS to indicate if startup process should output the highest available layer if the target output layer is not available.
- (RALS/O0139/Prop5) #9 Problem: It is asserted that if cross_layer_irap_aligned_flag is equal to 1 and two pictures having no dependency on each other in an access unit have different nal_unit_type values, the POC value alignment cannot be guaranteed. Decision (Ed): Agreed. The drafted intent was to enforce alignment by the flag only within each dependency tree. Editors to correct the text as necessary.
- (RALS/O0139/Prop1) #6 Proposal: Invoke the layer-wise start-up process for a base-layer CRA picture with HandleCraAsBlaFlag equal to 1. Decision (Ed): Check/clarify text as necessary if not already addressed (intent agreed in spirit).
- (RALS/O0220/Prop2) #5 Invoke the decoding process for generating unavailable reference pictures (subclause F.8.1.3) again when the current picture is the IRAP picture with NoRaslOutputFlag equal to 1. Decision (Ed BF): Check/clarify text as necessary if not already addressed (intent agreed in spirit).
- (RALS/O0220/Prop1 Alt2) #4 NoRaslOutputFlag is set to equal to 1 when the current picture is an IRAP picture, LayerInitializedFlag[k] = 0, and LayerInitializedFlag[refLayerId]=1 for all values of refLayerId equal to RefLayerId[k][j], where j is in the range of 0 to NumDirectRefLayers[k]-1, inclusive. In this solution, LayerInitializedFlag[k] is set equal to 1 after setting NoRaslOutputFlag to 1. Decision (Ed. BF): Adopted
- (RALS/O0149/Prop1) #10 Proposal: A base-layer IRAP picture that initiates the layer-wise start-up process (i.e. has NoClrasOutputFlag equal to 1) causes marking of all pictures in the DPB as “unused for reference”. Decision (Ed): Agreed.
- Rejected changes erroneously integrated under label (RALS/O0149/Prop2), since they are related to Prop1 and Prop3 of O0149.
- (Gen/O0137,O0200,O0223,Layer id) #32 Add (editorial equivalent of) "The value of nuh_layer_id shall be in the range of 0 to 62. The value of 63 for nuh_layer_id is reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore all data that follow the value 63 for nuh_layer_id in a NAL unit." and specify that vps_max_layers_minus1 shall not be equal to 63, but decoders shall allow that value to appear in the bitstream. Specify that the value 63 is interpreted the same as the value 62 (e.g., MaxLayersMinus1 = Min(62, vps_max_layers_minus1) and subsequently refer to MaxLayersMinus1 instead of vps_max_layers_minus1)

- (PS/O0109/default_one_target_output_layer_idc) #25 To change default_one_target_output_layer_flag to a two-bit default_one_target_output_layer_idc, and reserve the values 2 and 3
- (VUI/O0226/Mod tile WPP) #37 Modifications to the VUI indicators of tile and WPP alignment related syntax elements, from the r1.
- (Misc/O0062/POC LSB present condition) #31 Modification as decided later in trac.
- (RALS/O0149/Prop2): #11 Proposal: A new slice_reserved_flag is taken into use to indicate if a base-layer IDR picture initiates the layer-wise start-up process. Decision: Adopt (the bit should not be required to be present; if present should be the bit after the discardable_flag, and discardable_flag should be the first one of the three, and the poc reset flag is not required to be present).
- (Fix missing "!" before all_ref_layers_active_flag)
- (PSEM/O0142/Conditional extension syntax) #3 Adopt JCTVC-O0142 (as a structure to be used to switch whatever extensions we define in SPS, not necessarily committing to having these extensions be separate for each extension, but the current plan unless decided otherwise is to use one flag for range extensions syntax presence and one flag for SHVC+MV-HEVC extension syntax presence)
- (PS/O0118/visual signal info in vui per layer) #33 Add visual signal information (video_format, video_full_range_flag, colour_primaries, transfer_characteristics, matrix_coeffs) per layer to the VPS VUI, from v2 version of JCTVC-O0118.
- (Misc/O0062/POC LSB present) #31 The proposal's "option 3" is to add a flag in the VPS for each EL to control whether these LSBs are present or not (for IDR pictures), and when not present, the LSBs are inferred to be equal to 0. Decision: Adopted (as described herein).
- (ILDSD/O0225/signal max_tid_il_ref_pics per layer) #30 2nd proposal of JCTVC-O0225 regarding signalling of max_tid_il_ref_pics per layer, based upon relation to SCE2 on single loop decoding. Decision: Adopted.
- (ILDSD/O0225/max_tid_il_ref_pics RPL const.) #27 Change derivation of NumActiveRefLayerPics to consider max_tid_il_ref_pics.
- (ILDSD/O0120/sub_layers_vps_max_minus1 RPL const) #34 Modify inter-layer reference picture list default construction to incorporate max temporal sub-layers per layer syntax elements in VPS extension, from r2 version of JCTVC-O0120.
- (ILDSD/O0120/sub_layers_vps_max_minus1) #26 Add syntax elements to signal max temporal sub-layers for each layer in the VPS, with a gating flag, from JCTVC- option 2.
- (PS/O0223/Cross layer alignment type) #29 Add a flag in VPS VUI to indicate cross layer pic type alignment. Move cross_layer_irap_aligned_flag to VPS VUI and make presence condition on added flag
- (PS/O0092/Sharing SPS PPS) #17 Restrict sharing of SPS and PPS across layers to avoid creating problems during sub-bitstream extraction, based on modification of proposals in JCTVC-O0059 and JCTVC-O0092, reflected in the v2 version of O0092.
- (PS/FIX/N0092/Rep. Format) #19 Inferences.
- (PS/O0096/rep format) #20 Modify the SPS syntax for layers with nuh_layer_id > 0 to signal a reference to a rep_format index in the VPS, rather than signalling explicit representation format data in the SPS, from the v2 version of JCTVC-O0096.
- (PS/O0096/direct_dependency_type gating flag) #21 Add a gating flag in VPS extension to condition the presence of direct dependency type, with a default type signalled, from JCTVC-O0096
- (PS/O0109/view_id_len) #22 Modify the VPS extension syntax and semantics to replace view_id_len_minus1 with view_id_len, always signal that syntax element, add a constraint that (1<<view_id_len) >= NumViews, and modify view_id_val semantics to infer value of 0 when not present, from discussion of JCTVC-O0109
- (PS/O0109/profile_ref_minus1 constraint) #23 Modify the semantics of profile_ref_minus1[i] to replace “shall be less than i” with “shall be less than or equal to i”, from discussion of JCTVC-O0109
- (PS/O0109/vps_vui_present_flag move) #24 Move the vps_vui_present_flag to precede vps_vui_offset, and make vps_vui_offset conditional on that flag, from JCTVC-O0109
- (Fix misspelled LayerInitialisedFlag)
- (Fix MV-Trac Layers Present renaming) Incomplete renaming of “layers present” SEI.

Ed. Notes (Draft 5)

- --------- Release v5 -----------
- (Revised instructions) related to base spec.
- (Review GT04)
- (Review YK)
- (Removal unused variables)
- (Review GT03)
- (RA LSS CLA/E0306/TSA STSA align) If a higher layer pic is a TSA/STSA, lower layer inter-layer reference layer pictures in the same access unit shall also be TSA/STSA.

- (RA LSS CLA/E0306/picture marking) It was asked whether IDR/BLA in base layer but not in EL, the IDR in the BL causes marking of the EL pics as unused for reference (in other layers)? No, but need to figure out how/whether this is expressed in the text.
- (PS/N0085/Req. vui_timing_info_present_flag) No timing and HRD information in VUI for SPS with nuh_layer_id > 0: Require the flag in the SPS VUI to indicate that this data is not present.
- (Added VUI syntax to Annex F)
- (Comment E0102)
- (Moved F.8.3.4) Moved decoding process for reference picture lists construction to Annex G
- (Review GT02)
- (Review JC01)
- --------- Release v4 -----------
- (Review YK)
- ---------- Release v3 -----------
- (Revision scaling list)
- (Review JO01)
- (Review Editor's comments) Removed resolved and old comments, minor fixes related to scalability ids. constraints.
- (T PP/N0160/ilp_restricted_ref_layers_flag) #19 Move num_ilp_restricted_ref_layers and related offset delay syntax elements from SPS VUI to VPS VUI, and change to a num_ilp_restricted_ref_layers flag per direct dependent layer for each layer.
- (T PP/N0199/N0160/move tile boundaries alignment) #20 Adopt proposal 2 variant 2 (also in JCTVC-N0160) to move tile boundaries alignment flag from the SPS VUI to the VPS VUI, and also signal the flag per direct dependent layer for each layer.
- (PS/N0085/VPS VUI) #18 Add a VPS VUI section and put bit rate and picture rate information in it.
- (PS/N0085/ SPS,PPS IDs) #31 To establish that SPS/PPS IDs with different values of nuh_layer_id share the same "value space" such that different layers may share the same SPS/PPS. It is proposed to let them share the same value space.
- ----------- Release v2 -----------
- (Review CY02)
- (Review GT01) Simplification interlayer L0, L1 split
- (Review JB01)
- (GEN/JCTVC-N0244/POC), #27 Adopt to use a reserved slice header bit for a POC reset flag, plus signal POC LSB in enhancement layer IRAP pictures from JCTVC-N0065, to maintain POC alignment between layers when IRAP pictures are not aligned.[During joint session discussion, decided to align with (JCTVC-N0147/VPS IRAP aligned flag), and only require inclusion of the slice flag when the VPS alignment flag indicates non-aligned IRAPs possible.]
- (Review CY01)
- (RA LSS CLA/JCTVC-N0147/VPS IRAP aligned flag) #25 Add a flag in VPS extension to indicate if all IRAP pictures are aligned in set of dependent layers.
- (SEI/JCTVC-N0383/Add inter-layer constrained tile sets) #24
- (3D/E0038/View_id) #16 Adopt (merge with E0057). Revisions to integration of E0057
- (3D/Res. Constraint) #28 Support different spatial resolutions for different views but disable inter-view prediction in such a case.
- (3D/E0310/Levels) #29 Preliminary level definitions for stereo profile.
- (RA LSS CLA/JCTVC-N0066/layerwise startup) #26 Adopt version 2 layer-wise start up decoding process.
- (3D/E0240/3D reference display information SEI) #22 Persistence scope On 3D reference displays information SEI Decision
- ----------- Release v1 -----------
- (3D/E0057/ViewId) #16 Adopt (similar to E0038)
- (3D/E0104/Swap scalability dimensions) #15 Adopt, only portion that swaps multiview and depth flag in scalability dimension
- (3D/E0101/stereo profile avc_base_layer_flag) #14 Stereo profile definition the avc_base_flag which exists should be disabled.
- (TMVP COL//N0107/Col ind) #13 On collocated picture indication and inter_layer_sample_pred_only_flag) Remove the slice header syntax elements alt_collocated_indication_flag, collocated_ref_layer_idx, and inter_layer_sample_pred_only_flag.
- (RF/N0092/Rep. format information in VPS) #12 Adopt (with the u(4) adjustment)
- (SL ILP/N0120/max_tid_il_ref_pics_plus1_present_flag) #11 BoG Adopt with a minor editorial change to move location of inference.
- (RPLC/N0316/L0 L1 inter-layer rps) #10 BoG Exact decoding process might require slight modification based upon review of contributions related to view_id.

- (IL RPS/N0195/splitting_flag constraint) #9 Add constraint when splitting_flag is used, that the sum of the lengths be less than or equal to 6, from JCTVC-N0195 5th proposal.
- (SEI/N0173/Remove Layer Dependency SEI) #8 Layer dependency change SEI message be removed from specification. If the SEI message does remain, to adopt JCTVC-N0174 (with some editorial improvements).
- (IL RPS/N0195/ilp_slice_signaling_flag) #7 Adopt an Inter Layer Reference Picture (ILRP) presence flag in the VPS, conditioning the presence of ILRP syntax elements in the slice segment header, similar to JCTVC-N0195 proposal 2.
- (IL RPS/N0081,N0195,N0154,N0217/inter_pred_layer_idc) #6 Adopt a condition on signalling inter_layer_pred_layer_idc[i], to avoid sending when NumDirectRefLayers equals NumActiveRefLayerPics, and instead infer values.
- (Incl. PPS RBSP syntax) Included from base spec.
- (EPSPS/N0371/Scaling list prediction) #5 Adopting scaling list prediction in SPS and PPS (harmonization of JCTVC-N0162 and JCTVC-N0200 variant 3)
- (RPLC/N0082/Init RPL) #4 The BoG recommends adopting initialization process of reference picture lists.
- (PS/N0085/Editorial suggestions) #3 Editorial changes – delegated to editors for consideration.
- (PS/N0085/vps nuh_layer_id) #2 Add a restriction "The value of nuh_layer_id of a VPS NAL unit shall be equal to 0." (for bitstreams conforming to specified proposals, and decoder shall ignore VPS NALUs with other values of nuh_layer_id.
- (PS/N0085/vps_extension_offset) #1 Semantics of vps_extension_offset: It is proposed to clearly specify that emulation prevention bytes are counted.

Ed. Notes (JCT3V-E0100)

- (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)

Remarks:

- Modifications in long sections copied from the HEVC spec are highlighted in turquoise. Open issues and editor's notes are highlighted in yellow.
- Incorporation of JCT3V-E0102 has been deferred since further coordination with JCT-VC on auxiliary picture structure and constraints is needed.

CONTENTS

Add the following definitions to clause 3:

3.X **base bitstream partition**: A *bitstream partition* that is also a conforming *bitstream* itself.

3.X **bitstream partition**: A sequence of bits, in the form of a *NAL unit stream* or a *byte stream*, that is a subset of a *bitstream* according to a *partitioning*.

3.X **output layer set**: A *layer set* that is associated with a set of target output layers.

Replace the definition of operation point in clause 3 with the following:

3.X **operation point**: A bitstream that is created from another *bitstream* by operation of the *sub-bitstream extraction process* with the another *bitstream*, a target highest TemporalId, and a target *layer identifier list* as inputs, and that is associated with a set of target output layers.

NOTE 14 – If the target highest TemporalId of an operation point is equal to the greatest value of TemporalId in the layer set associated with the target layer identification list, the operation point is identical to the layer set. Otherwise it is a subset of the layer set.

Replace subclauses 7.4.2.4.2 with the following (with differences indicated in turquois):

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

This subclause specifies the activation process of VPSs, SPSs, and PPSs.

NOTE 1 – The VPS, SPS, and PPS mechanism decouples the transmission of infrequently changing information from the transmission of coded block data. VPSs, SPSs, and PPSs may, in some applications, be conveyed "out-of-band".

A PPS RBSP includes parameters that can be referred to by the coded slice segment NAL units of one or more coded pictures. Each PPS RBSP is initially considered not active for any layer at the start of the operation of the decoding process. At most one PPS RBSP is considered active for each layer at any given moment during the operation of the decoding process, and the activation of any particular PPS RBSP for a particular layer results in the deactivation of the previously-active PPS RBSP for the particular layer (if any).

One PPS RBSP may be the active PPS RBSP for more than one layer. When not explicitly specified, the layer a PPS RBSP is active for is inferred to be the current layer in the context where the active PPS RBSP is referred to.

When a PPS RBSP (with a particular value of pps_pic_parameter_set_id) is not active for a particular layer 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 value) of the particular layer, it is activated for the particular layer. This PPS RBSP is called the active PPS RBSP for the particular layer until it is deactivated by the activation of another PPS RBSP for the particular 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.

Any PPS NAL unit containing the value of pps_pic_parameter_set_id for the active PPS RBSP for a coded picture (and consequently for the layer containing the coded picture) shall have the same content as that of the active 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.

An SPS RBSP includes parameters that can be referred to by one or more PPS RBSPs or one or more SEI NAL units containing an active parameter sets SEI message. Each SPS RBSP is initially considered not active for any layer at the start of the operation of the decoding process. At most one SPS RBSP is considered active for each layer at any given moment during the operation of the decoding process, and the activation of any particular SPS RBSP for a particular layer results in the deactivation of the previously-active SPS RBSP for the particular layer value of nuh_layer_id (if any).

One SPS RBSP may be the active SPS RBSP for more than one layer. When not explicitly specified, the layer an SPS RBSP is active for is inferred to be the current layer in the context where the active PPS RBSP is referred to.

When an SPS RBSP (with a particular value of sps_seq_parameter_set_id) is not already active for a particular layer and it is referred to by activation of a PPS RBSP (in which pps_seq_parameter_set_id is equal to the sps_seq_parameter_set_id value) referred to by the particular layer or is referred to by an SEI NAL unit containing an active parameter sets SEI message (in which one of the active_seq_parameter_set_id[i] values is equal to the sps_seq_parameter_set_id value), it is activated for the particular layer. This SPS RBSP is called the active SPS RBSP for the particular layer until it is deactivated by the activation of another SPS RBSP for the particular 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 the base layer shall remain active for the entire CVS.

NOTE 2 – Because an IRAP access unit with NoRaslOutputFlag equal to 1 begins a new CVS and an activated SPS RBSP must remain active for the entire CVS, an SPS RBSP can only be activated by an active parameter sets SEI message when the active parameter sets SEI message is part of an IRAP access unit with NoRaslOutputFlag equal to 1.

Any SPS NAL unit containing the value of sps_seq_parameter_set_id for the active SPS RBSP for the base layer for a CVS shall have the same content as that of the active SPS RBSP for the base layer for the CVS, unless it follows the last access unit of the CVS and precedes the first VCL NAL unit and the first SEI NAL unit containing an active parameter sets SEI message (when present) of another CVS.

A VPS RBSP includes parameters that can be referred to by one or more SPS RBSPs or one or more SEI NAL units containing an active parameter sets SEI message. Each VPS RBSP is initially considered not active at the start of the operation of the decoding process. At most one VPS RBSP is considered active at any given moment during the operation of the decoding process, and the activation of any particular VPS RBSP results in the deactivation of the previously-active VPS RBSP (if any).

When a VPS RBSP (with a particular value of vps_video_parameter_set_id) is not already active and it is referred to by activation of an SPS RBSP (in which sps_video_parameter_set_id is equal to the vps_video_parameter_set_id value), or is referred to by an SEI NAL unit containing an active parameter sets SEI message (in which active_video_parameter_set_id is equal to the vps_video_parameter_set_id value), it is activated. This VPS RBSP is called the active VPS RBSP until it is deactivated by the activation of another VPS RBSP. A VPS RBSP, with that particular value of vps_video_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 VPS RBSP shall remain active for the entire CVS.

NOTE 3 – Because an IRAP access unit with NoRaslOutputFlag equal to 1 begins a new CVS and an activated VPS RBSP must remain active for the entire CVS, a VPS RBSP can only be activated by an active parameter sets SEI message when the active parameter sets SEI message is part of an IRAP access unit with NoRaslOutputFlag equal to 1.

Any VPS NAL unit containing the value of vps_video_parameter_set_id for the active VPS RBSP for a CVS shall have the same content as that of the active VPS RBSP for the CVS, unless it follows the last access unit of the CVS and precedes the first VCL NAL unit, the first SPS NAL unit, and the first SEI NAL unit containing an active parameter sets SEI message (when present) of another CVS.

NOTE 4 – If VPS RBSP, SPS RBSP, or PPS RBSP are conveyed within the bitstream, these constraints impose an order constraint on the NAL units that contain the VPS RBSP, SPS RBSP, or PPS RBSP, respectively. Otherwise (VPS RBSP, SPS RBSP, or PPS RBSP are conveyed by other means not specified in this Specification), they must be available to the decoding process in a timely fashion such that these constraints are obeyed.

All constraints that are expressed on the relationship between the values of the syntax elements and the values of variables derived from those syntax elements in VPSs, SPSs, and PPSs and other syntax elements are expressions of constraints that apply only to the active VPS RBSP, the active SPS RBSP for the base layer, and the active PPS RBSP for the base layer. If any VPS RBSP, SPS RBSP, and PPS RBSP is present that is never activated in the bitstream, its syntax elements shall have values that would conform to the specified constraints if it was activated by reference in an otherwise conforming bitstream.

During operation of the decoding process (see clause 8), the values of parameters of the active VPS, the active SPS for the base layer, and the active PPS RBSP for the base layer are considered in effect. For interpretation of SEI messages, the values of the active VPS RBSP, the active SPS RBSP for the base layer, and the active PPS RBSP for the base layer for the operation of the decoding process for the VCL NAL units of the coded picture in the same access unit are considered in effect unless otherwise specified in the SEI message semantics.

Replace clause 8, subclauses 8.1, 8.2, 8.3, 8.3.2, 8.3.3, and 8.3.3.1 with the following and add subclause 8.1.1 (with differences indicated in turquois):

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 variable TargetOptLayerSetIdx, which specifies the index to the list of the output layer sets specified by the VPS, of the target output layer set, is specified as follows:

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

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

– Otherwise, TargetOptLayerSetIdx is set equal to 0.

The variable TargetDecLayerSetIdx, the layer identifier list TargetOptLayerIdList, which specifies the list of nuh_layer_id values, in increasing order of nuh_layer_id values, of the pictures to be output, and 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, are specified as follows:

```
TargetDecLayerSetIdx = output_layer_set_idx_minus1[ TargetOptLayerSetIdx ] + 1
lsIdx = TargetDecLayerSetIdx
for( k = 0, j = 0; j < NumLayersInIdList[ lsIdx ]; j++ ) {
    TargetDecLayerIdList[ j ] = LayerSetLayerIdList[ lsIdx ][ j ]                    (8-1)
    if( output_layer_flag[ lsIdx ][ j ] )
        TargetOptLayerIdList[ k++ ] = LayerSetLayerIdList[ lsIdx ][ j ]
}
```

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).

When the current picture is an IRAP picture, the variable HandleCraAsBlaFlag is derived as specified in the following:

– 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.

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

When the current picture is an IRAP picture and has nuh_layer_id equal to 0, the following applies:

– The variable NoClrasOutputFlag is specified as follows:

  – If the current picture is the first picture in the bitstream, NoClrasOutputFlag is set equal to 1.

  – Otherwise, if the current picture is a BLA picture or a CRA with HandleCraAsBlaFlag equal to 1, NoClrasOutputFlag is set equal to 1.

  – Otherwise, if the current picture is an IDR picture and cross_layer_bla_flag is equal to1, NoClrasOutputFlag is set equal to 1.

  – Otherwise, if some external means not specified in this Specification is available to set NoClrasOutputFlag, NoClrasOutputFlag is set by the external means.

  – Otherwise, NoClrasOutputFlag is set equal to 0.

– When NoClrasOutputFlag is equal to 1, the variable LayerInitializedFlag[i] is set equal to 0 for all values of i from 0 to vps_max_layer_id, inclusive, and the variable FirstPicInLayerDecodedFlag[i] is set equal to 0 for all values of i from 1 to vps_max_layer_id, inclusive.

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

– Otherwise, if LayerInitializedFlag[nuh_layer_id] is equal to 0 and LayerInitializedFlag[refLayerId] is equal to 1 for all values of refLayerId equal to RefLayerId[nuh_layer_id][j], where j is in the range of 0 to NumDirectRefLayers[nuh_layer_id] – 1, inclusive, the variable NoRaslOutputFlag is set equal to 1.

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

When the current picture is an IRAP picture with NoRaslOutputFlag equal to 1 and one of the following conditions is true, LayerInitializedFlag[nuh_layer_id] is set equal to 1:

– nuh_layer_id is equal to 0.

– LayerInitializedFlag[nuh_layer_id] is equal to 0 and LayerInitializedFlag[refLayerId] is equal to 1 for all values of refLayerId equal to RefLayerId[nuh_layer_id][j], where j is in the range of 0 to NumDirectRefLayers[nuh_layer_id] – 1, inclusive.

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 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 – 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).

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 as specified in subclause 8.1.1 is invoked.

8.1.1 Decoding process for a coded picture with nuh_layer_id 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 VCL NAL units of the current picture and their associated non-VCL NAL units.

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 parses the RBSP syntax structure.

8.3 Slice decoding process

[Ed. (CY): consider moving the remaining part of 8.3, the entire 8.1 and entire 8.2 to Annex F. To be confirmed before the action is taken.]

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 8.3.4. 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".

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).

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

When the current picture is an IRAP picture with nuh_layer_id equal to 0 and NoClrasOutputFlag is equal to 1, all reference pictures with any value of nuh_layer_id currently in the DPB (if any) are marked as "unused for reference".

When the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, all reference pictures with nuh_layer_id equal to currPicLayerId 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                                                              (8-2)
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 −
                       PicOrderCntVal & ( MaxPicOrderCntLsb − 1 )
    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 for the current layer 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 for the current layer. 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 NumPicTotalCurr 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 NumPicTotalCurr:

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

– Otherwise, when the current picture contains a P or B slice, the value of NumPicTotalCurr 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 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 PicOrderCntVal & ( MaxPicOrderCntLsb − 1 )
                                    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"                    (8-3)
for( i = 0; i < NumPocLtFoll; i++ )
    if( !FollDeltaPocMsbPresentFlag[ i ] )
        if( there is a reference picture picX in the DPB with PicOrderCntVal & ( MaxPicOrderCntLsb − 1 )
                        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 or RefPicSetLtFoll and have 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"                   (8-4)

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 have 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 clause 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 clause 8.3.3

  – A picture that precedes the associated IRAP picture in decoding order

– When 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.

8.3.3 Decoding process for generating unavailable reference pictures

8.3.3.1 General decoding process for generating unavailable reference pictures

This process is invoked once per coded picture when the current picture is a BLA picture or is a CRA picture with NoRaslOutputFlag equal to 1.

NOTE – This process is primarily specified only for the specification of syntax constraints for RASL pictures. The entire specification of the decoding process for RASL pictures associated with an IRAP picture that has NoRaslOutputFlag equal to 1 is included herein only for purposes of specifying constraints on the allowed syntax content of such RASL pictures. During the decoding process, any RASL pictures associated with an IRAP picture that has NoRaslOutputFlag equal to 1 may be ignored, as these pictures are not specified for output and have no effect on the decoding process of any other pictures that are specified for output. However, in HRD operations as specified in Annex C, RASL access units may need to be taken into consideration in derivation of CPB arrival and removal times.

When this process is invoked, the following applies:

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

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

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

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

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

  – The value of nuh_layer_id for the generated picture is inferred to be equal to nuh_layer_id.

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

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

  – The value of slice_pic_order_cnt_lsb for the generated picture is inferred to be equal to (PocLtFoll[i] & (MaxPicOrderCntLsb – 1)).

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

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

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

– The value of nuh_layer_id for the generated picture is inferred to be equal to nuh_layer_id.

Replace Annex C with the following (with differences indicated in turquois):

Annex C

Hypothetical reference decoder

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

C.1 General

This annex specifies the hypothetical reference decoder (HRD) and its use to check bitstream and decoder conformance.

Two types of bitstreams or bitstream subsets are subject to HRD conformance checking for this Specification. The first type, called a Type I bitstream, is a NAL unit stream containing only the VCL NAL units and NAL units with nal_unit_type equal to FD_NUT (filler data NAL units) for all access units in the bitstream. The second type, called a Type II bitstream, contains, in addition to the VCL NAL units and filler data NAL units for all access units in the bitstream, at least one of the following:

– additional non-VCL NAL units other than filler data NAL units,

– all leading_zero_8bits, zero_byte, start_code_prefix_one_3bytes, and trailing_zero_8bits syntax elements that form a byte stream from the NAL unit stream (as specified in Annex B).

Figure C-1 shows the types of bitstream conformance points checked by the HRD.

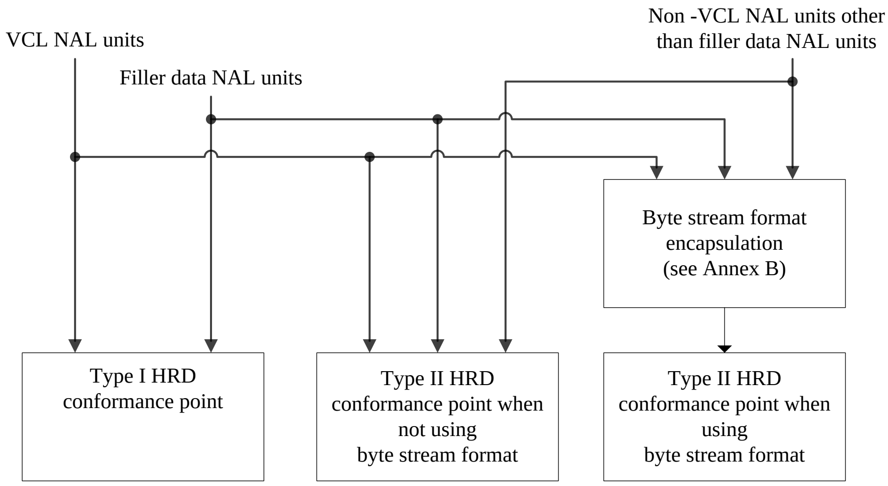

Figure C-1 – Structure of byte streams and NAL unit streams for HRD conformance checks

The syntax elements of non-VCL NAL units (or their default values for some of the syntax elements), required for the HRD, are specified in the semantic subclauses of clause 7, Annexes D and E.

Two types of HRD parameter sets (NAL HRD parameters and VCL HRD parameters) are used. The HRD parameter sets are signalled through the hrd_parameters( ) syntax structure, which may be part of the SPS syntax structure or the VPS syntax structure.

Multiple tests may be needed for checking the conformance of a bitstream, which is referred to as the bitstream under test. For each test, the following steps apply in the order listed:

1. An operation point under test, denoted as TargetOp, is selected by selecting a target output layer set identified by TargetOptLayerSetIdx and selecting a target highest TemporalId value HighestTid. The value of TargetOptLayerSetIdx shall be in the range of 0 to NumOutputLayerSets − 1, inclusive. The value of HighestTid shall be in the range of 0 to vps_max_sub_layers_minus1, inclusive. The variables TargetDecLayerSetIdx, TargetOptLayerIdList, and TargetDecLayerIdList are then derived as specified by Equation 8-1. The operation point under test has OptLayerIdList equal to TargetOptLayerIdList, OpLayerIdList equal to TargetDecLayerIdList, and OpTid equal to HighestTid.

2. The sub-bitstream extraction process as specified in clause 10 is invoked with the bitstream under test, HighestTid, and TargetDecLayerIdList as inputs, and the output is assigned to BitstreamToDecode.

3. When both the vps_vui_bsp_hrd_parameters( ) syntax structure is present in the active VPS and num_bitstream_partitions[TargetDecLayerSetIdx] is greater than 1 or both a bitstream partition HRD parameters SEI message is present and the SEI message contains syntax element sei_num_bitstream_partitions_minus1[TargetDecLayerSetIdx] greater than 0, either the bitstream-specific CPB operation or the bitstream-partition-specific CPB operation is selected for a conformance test, and both CPB operations shall be tested for checking the conformance of a bitstream. When the bitstream-specific CPB operation is tested, the subsequent steps apply for the bitstream under test. When the bitstream-partition-specific CPB operation is tested, the subsequent steps apply to each bitstream partition of the bitstream under test, referred to as the bitstream partition under test. When the bitstream-partition-specific CPB operation is tested and the input to the HRD is a bitstream, the bitstream partitions are derived with the demultiplexing process for deriving a bitstream partition in subclause C.6.

4. The hrd_parameters( ) syntax structure and the sub_layer_hrd_parameters( ) syntax structure applicable to TargetOp are selected as follows:

   – If the bitstream-specific CPB operation is tested, the following applies:

     – If TargetDecLayerIdList contains all nuh_layer_id values present in the bitstream under test, the hrd_parameters( ) syntax structure in the active SPS for the base layer (or provided through an external means not specified in this Specification) is selected.

     – Otherwise, the hrd_parameters( ) syntax structure in the active VPS (or provided through some external means not specified in this Specification) that applies to TargetOp is selected.

   – Otherwise, the hrd_parameters( ) syntax structure is selected as follows:

     – Either one of the hrd_parameters( ) syntax structures in the following conditions can be selected, if both of the following conditions are true:

       – The vps_vui_bsp_hrd_parameters( ) syntax structure is present in the active VPS (or is available through some external means not specified in this Specification) and contains a hrd_parameters( ) syntax structure that applies to TargetOp and to the bitstream partition under test.

       – A bitstream partition HRD parameters SEI message that is included in a scalable nesting SEI message that applies to TargetOp and contains a hrd_parameters( ) syntax structure that applies to TargetOp and to the bitstream partition under test is present (or is available through some external means not specified in this Specification).

     – Otherwise, if the vps_vui_bsp_hrd_parameters( ) syntax structure is present in the active VPS (or is available through some external means not specified in this Specification) and contains a hrd_parameters( ) syntax structure that applies to TargetOp and the bitstream partition under test, that hrd_parameters( ) syntax structure is selected.

     – Otherwise, a hrd_parameters( ) syntax structure that applies to the bitstream partition under test in the bitstream partition HRD parameters SEI message that is included in a scalable nesting SEI message that applies to TargetOp shall be present (or shall be available through some external means not specified in this Specification) and is selected.

   Within the selected hrd_parameters( ) syntax structure, if BitstreamToDecode is a Type I bitstream, the sub_layer_hrd_parameters(HighestTid) syntax structure that immediately follows the condition "if(vcl_hrd_parameters_present_flag)" is selected and the variable NalHrdModeFlag is set equal to 0; otherwise (BitstreamToDecode is a Type II bitstream), the sub_layer_hrd_parameters(HighestTid) syntax structure that immediately follows either the condition "if(vcl_hrd_parameters_present_flag)" (in this case the variable NalHrdModeFlag is set equal to 0) or the condition "if(nal_hrd_parameters_present_flag)" (in this case the variable NalHrdModeFlag is set equal to 1) is selected. When BitstreamToDecode is a Type II bitstream and NalHrdModeFlag is equal to 0, all non-VCL NAL units except filler data NAL units, and all leading_zero_8bits, zero_byte, start_code_prefix_one_3bytes, and trailing_zero_8bits syntax elements that form a byte stream from the NAL unit stream (as specified in Annex B), when present, are discarded from BitstreamToDecode, and the remaining bitstream is assigned to BitstreamToDecode.

5. An access unit associated with a buffering period SEI message (present in BitstreamToDecode or available through external means not specified in this Specification) applicable to TargetOp is selected as the HRD initialization point and referred to as access unit 0. An applicable buffering period SEI message is available through external means not specified in this Specification or is selected from access unit 0 as follows:

   – If the bitstream-specific CPB operation is tested, the following applies:

     – If TargetDecLayerIdList contains all nuh_layer_id values present in the bitstream under test, a non-nested buffering period SEI message is selected.

     – Otherwise, a buffering period SEI message included in the scalable nesting SEI message with bitstream_subset_flag equal to 1 and applicable to TargetOp is selected.

– Otherwise, a buffering period SEI message included in the bitstream partition nesting SEI message applicable to the bitstream partition under test is selected.

6. For each access unit in BitstreamToDecode starting from access unit 0, the buffering period SEI message (present in BitstreamToDecode or available through external means not specified in this Specification) that is associated with the access unit and applies to TargetOp is selected, the picture timing SEI message (present in BitstreamToDecode or available through external means not specified in this Specification) that is associated with the access unit and applies to TargetOp is selected, and when SubPicHrdFlag is equal to 1 and sub_pic_cpb_params_in_pic_timing_sei_flag is equal to 0, the decoding unit information SEI messages (present in BitstreamToDecode or available through external means not specified in this Specification) that are associated with decoding units in the access unit and apply to TargetOp are selected as follows:

   – If the bitstream-specific CPB operation is tested, the following applies:

     – If TargetDecLayerIdList contains all nuh_layer_id values present in the bitstream under test, non-nested buffering period, picture timing and decoding unit information SEI messages are selected.

     – Otherwise, buffering period, picture timing and decoding unit information SEI messages included in the scalable nesting SEI message with bitstream_subset_flag equal to 1 and applicable to TargetOp are selected.

   – Otherwise, buffering period, picture timing and decoding unit information SEI messages included in the bitstream partition nesting SEI message and applicable to the bitstream partition under test are selected.

7. A value of SchedSelIdx is selected as follows:

   – If the bitstream-specific CPB operation is tested, the selected SchedSelIdx shall be in the range of 0 to cpb_cnt_minus1[HighestTid], inclusive, where cpb_cnt_minus1[HighestTid] is found in the sub_layer_hrd_parameters(HighestTid) syntax structure as selected above.

   – Otherwise (the bitstream-partition-specific CPB operation is tested), a SchedSelCombIdx is selected for the bitstream under test and used for each bitstream partition under test. The following applies:

     – If the vps_vui_bsp_hrd_parameters( ) syntax structure is present in the active VPS (or made available through external means not specified in this Specification) and contains the selected hrd_parameters( ) syntax structure that applies to TargetOp and the bitstream partition under test, the selected SchedSelCombIdx shall be in the range of 0 to num_bsp_sched_combinations[TargetDecLayerSetIdx] − 1, inclusive. and the selected SchedSelIdx shall be equal to bsp_comb_sched_idx[TargetDecLayerSetIdx][SchedSelCombIdx][j] where j is the index of the bitstream partition under test.

     – Otherwise, the selected SchedSelCombIdx shall be in the range of 0 to sei_num_bsp_sched_combinations_minus1[TargetDecLayerSetIdx], inclusive. and the selected SchedSelIdx shall be equal to sei_bsp_comb_sched_idx[TargetDecLayerSetIdx][SchedSelCombIdx][j] of the bitstream partition HRD parameters SEI message applicable to TargetOp where j is the index of the bitstream partition under test.

8. When the coded picture in access unit 0 has nal_unit_type equal to CRA_NUT or BLA_W_LP, and irap_cpb_params_present_flag in the selected buffering period SEI message is equal to 1, either of the following applies for selection of the initial CPB removal delay and delay offset:

   – If NalHrdModeFlag is equal to 1, the default initial CPB removal delay and delay offset represented by nal_initial_cpb_removal_delay[SchedSelIdx] and nal_initial_cpb_removal_offset[SchedSelIdx], respectively, in the selected buffering period SEI message are selected. Otherwise, the default initial CPB removal delay and delay offset represented by vcl_initial_cpb_removal_delay[SchedSelIdx] and vcl_initial_cpb_removal_offset[SchedSelIdx], respectively, in the selected buffering period SEI message are selected. The variable DefaultInitCpbParamsFlag is set equal to 1.

   – If NalHrdModeFlag is equal to 1, the alternative initial CPB removal delay and delay offset represented by nal_initial_alt_cpb_removal_delay[SchedSelIdx] and nal_initial_alt_cpb_removal_offset[SchedSelIdx], respectively, in the selected buffering period SEI message are selected. Otherwise, the alternative initial CPB removal delay and delay offset represented by vcl_initial_alt_cpb_removal_delay[SchedSelIdx] and vcl_initial_alt_cpb_removal_offset[SchedSelIdx], respectively, in the selected buffering period SEI message are selected. The variable DefaultInitCpbParamsFlag is set equal to 0, and the RASL access units associated with access unit 0 are discarded from BitstreamToDecode and the remaining bitstream is assigned to BitstreamToDecode.

9. For the bitstream-partition-specific CPB operation, SubPicHrdFlag is set equal to 1. For the bitstream-specific CPB operation, when sub_pic_hrd_params_present_flag in the selected hrd_parameters( ) syntax structure is equal to 1, the CPB is scheduled to operate either at the access unit level (in which case the variable

SubPicHrdFlag is set equal to 0) or at the sub-picture level (in which case the variable SubPicHrdFlag is set equal to 1).

For each operation point under test when the bitstream-specific CPB operation is tested, the number of bitstream conformance tests to be performed is equal to n0 * n1 * (n2 * 2 + n3) * n4, where the values of n0, n1, n2, n3, and n4 are specified as follows:

– n0 is derived as follows:
  – If BitstreamToDecode is a Type I bitstream, n0 is equal to 1.
  – Otherwise (BitstreamToDecode is a Type II bitstream), n0 is equal to 2.
– n1 is equal to cpb_cnt_minus1[HighestTid] + 1.
– n2 is the number of access units in BitstreamToDecode that each is associated with a buffering period SEI message applicable to TargetOp and for each of which both of the following conditions are true:
  – nal_unit_type is equal to CRA_NUT or BLA_W_LP for the VCL NAL units;
  – The associated buffering period SEI message applicable to TargetOp has irap_cpb_params_present_flag equal to 1.
– n3 is the number of access units in BitstreamToDecode BitstreamToDecode that each is associated with a buffering period SEI message applicable to TargetOp and for each of which one or both of the following conditions are true:
  – nal_unit_type is equal to neither CRA_NUT nor BLA_W_LP for the VCL NAL units;
  – The associated buffering period SEI message applicable to TargetOp has irap_cpb_params_present_flag equal to 0.
– n4 is derived as follows:
  – If sub_pic_hrd_params_present_flag in the selected hrd_parameters( ) syntax structure is equal to 0, n4 is equal to 1;
  – Otherwise, n4 is equal to 2.

When BitstreamToDecode is a Type II bitstream, the following applies:

– If the sub_layer_hrd_parameters(HighestTid) syntax structure that immediately follows the condition "if(vcl_hrd_parameters_present_flag)" is selected, the test is conducted at the Type I conformance point shown in Figure C-1, and only VCL and filler data NAL units are counted for the input bit rate and CPB storage.
– Otherwise (the sub_layer_hrd_parameters(HighestTid) syntax structure that immediately follows the condition "if(nal_hrd_parameters_present_flag)" is selected), the test is conducted at the Type II conformance point shown in Figure C-1, and all bytes of the Type II bitstream, which may be a NAL unit stream or a byte stream, are counted for the input bit rate and CPB storage.

NOTE 1 – NAL HRD parameters established by a value of SchedSelIdx for the Type II conformance point shown in Figure C-1 are sufficient to also establish VCL HRD conformance for the Type I conformance point shown in Figure C-1 for the same values of InitCpbRemovalDelay[SchedSelIdx], BitRate[SchedSelIdx], and CpbSize[SchedSelIdx] for the VBR case (cbr_flag[SchedSelIdx] equal to 0). This is because the data flow into the Type I conformance point is a subset of the data flow into the Type II conformance point and because, for the VBR case, the CPB is allowed to become empty and stay empty until the time a next picture is scheduled to begin to arrive. For example, when decoding a CVS conforming to one or more of the profiles specified in Annex A using the decoding process specified in clauses 2 through 10, when NAL HRD parameters are provided for the Type II conformance point that not only fall within the bounds set for NAL HRD parameters for profile conformance in item f) of subclause A.4.2 but also fall within the bounds set for VCL HRD parameters for profile conformance in item e) of subclause A.4.2, conformance of the VCL HRD for the Type I conformance point is also assured to fall within the bounds of item e) of subclause A.4.2.

All VPSs, SPSs and PPSs referred to in the VCL NAL units, and the corresponding buffering period, picture timing and decoding unit information SEI messages shall be conveyed to the HRD, in a timely manner, either in the bitstream (by non-VCL NAL units), or by other means not specified in this Specification.

In Annexes C, D, and E, the specification for "presence" of non-VCL NAL units that contain VPSs, SPSs, PPSs, buffering period SEI messages, picture timing SEI messages, or decoding unit information SEI messages is also satisfied when those NAL units (or just some of them) are conveyed to decoders (or to the HRD) by other means not specified in this Specification. For the purpose of counting bits, only the appropriate bits that are actually present in the bitstream are counted.

NOTE 2 – As an example, synchronization of such a non-VCL NAL unit, conveyed by means other than presence in the bitstream, with the NAL units that are present in the bitstream, can be achieved by indicating two points in the bitstream, between which the non-VCL NAL unit would have been present in the bitstream, had the encoder decided to convey it in the bitstream.

When the content of such a non-VCL NAL unit is conveyed for the application by some means other than presence within the bitstream, the representation of the content of the non-VCL NAL unit is not required to use the same syntax as specified in this Specification.

NOTE 3 – When HRD information is contained within the bitstream, it is possible to verify the conformance of a bitstream to the requirements of this subclause based solely on information contained in the bitstream. When the HRD information is not present in the bitstream, as is the case for all "stand-alone" Type I bitstreams, conformance can only be verified when the HRD data are supplied by some other means not specified in this Specification.

For the bitstream-specific CPB operation, the HRD contains a coded picture buffer (CPB), an instantaneous decoding process, a decoded picture buffer (DPB) that contains a sub-DPB for each layer, and output cropping as shown in Figure C-2.

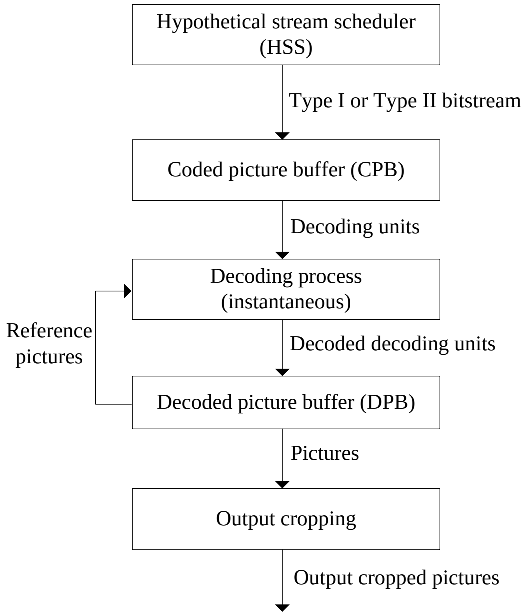

Figure C-2 – Bitstream-specific HRD buffer model

For the bitstream-partition-specific CPB operation, the HRD contains a bitstream demultiplexer (optionally present), two or more bitstream partition buffers (BPB), two or more instantaneous decoding processes, a decoded picture buffer (DPB) that contains a sub-DPB for each layer, and output cropping as shown in Figure C-3.

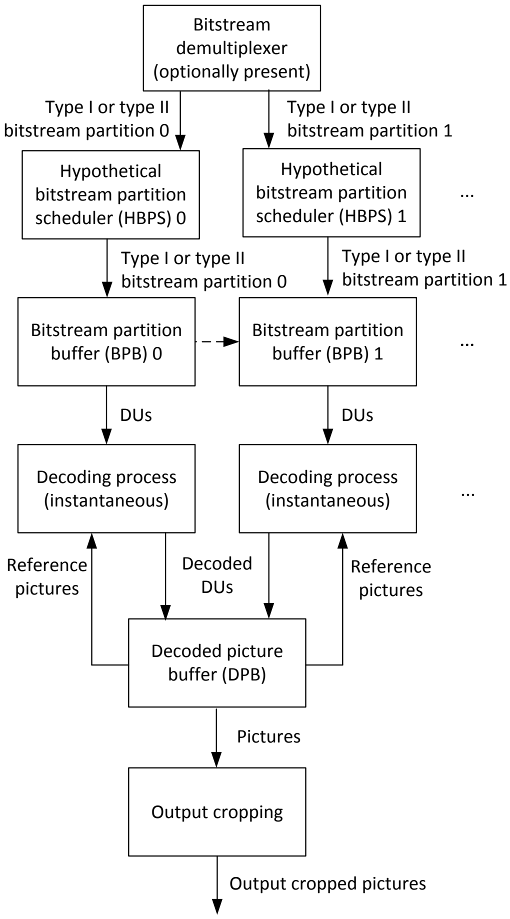

Figure C-3 – Bitstream-partition-specific HRD buffer model

For each bitstream conformance test, the CPB size (number of bits) for the bitstream-specific CPB operation and the BPB size for the bitstream-partition-specific CPB operation is CpbSize[SchedSelIdx] as specified in subclause E.3.3, where SchedSelIdx and the HRD parameters are specified above in this subclause. When a CVS conforming to one or more of the profiles specified in Annex A is decoded by applying the decoding process specified in clauses 2−10, the sub-DPB size (number of picture storage buffers) of the sub-DPB for the base layer is sps_max_dec_pic_buffering_minus1[HighestTid] + 1, where sps_max_dec_pic_buffering_minus1[HighestTid] is from the active SPS for the base layer. When a CVS conforming to one or more of the profiles specified in Annex G or H is decoded by applying the decoding process specified in clauses 2−10, Annex F, and Annex G or H, the sub-DPB size of the sub-DPB for a layer with nuh_layer_id equal to currLayerId is max_vps_dec_pic_buffering_minus1[TargetOptLayerSetIdx][currLayerId][HighestTid] + 1, where max_vps_dec_pic_buffering_minus1[TargetOptLayerSetIdx][currLayerId][HighestTid] + 1 is from the active VPS.

The variable SubPicHrdPreferredFlag is either specified by external means, or when not specified by external means, set equal to 0.

When the value of the variable SubPicHrdFlag has not been set by step 9 above in this subclause, it is derived as follows:

$$\text{SubPicHrdFlag} = \text{SubPicHrdPreferredFlag} \;\&\&\; \text{sub\_pic\_hrd\_params\_present\_flag} \qquad \text{(C-1)}$$

If SubPicHrdFlag is equal to 0, the HRD operates at access unit level and each decoding unit is an access unit. Otherwise the HRD operates at sub-picture level and each decoding unit is a subset of an access unit.

NOTE 4 – If the HRD operates at access unit level, each time a decoding unit that is an entire access unit is removed from the CPB. Otherwise (the HRD operates at sub-picture level), each time a decoding unit that is a subset of an access unit is removed from the CPB. In both cases, each time an entire decoded picture is output from the DPB, though the picture output time is derived based on the differently derived CPB removal times and the differently signalled DPB output delays.

The following is specified for expressing the constraints in this annex:

– Each access unit is referred to as access unit n, where the number n identifies the particular access unit. Access unit 0 is selected per step 5 above. The value of n is incremented by 1 for each subsequent access unit in decoding order.

– Each decoding unit is referred to as decoding unit m, where the number m identifies the particular decoding unit. The first decoding unit in decoding order in access unit 0 is referred to as decoding unit 0. The value of m is incremented by 1 for each subsequent decoding unit in decoding order.

NOTE 5 – The numbering of decoding units is relative to the first decoding unit in access unit 0.

– Picture n refers to the coded picture or the decoded picture of access unit n.

The HRD operates as follows:

– The HRD is initialized at decoding unit 0, with the CPB, each sub-DPB of the DPB, and each BPB being set to be empty (the sub-DPB fullness for each sub-DPB is set equal to 0).

NOTE 6 – After initialization, the HRD is not initialized again by subsequent buffering period SEI messages.

– For the bitstream-specific CPB operation, data associated with decoding units that flow into the CPB according to a specified arrival schedule are delivered by the HSS. For the bitstream-partition-specific CPB operation, data associated with decoding units that flow into the BPB according to a specified arrival schedule are delivered by an HBPS.

– When the bitstream-partition-specific CPB operation is used, each bitstream partition with index j is processed as specified in clause C.2 with the HSS replaced by the HPBS and with SchedSelIdx equal to bsp_comb_sched_idx[TargetDecLayerSetIdx][SchedSelCombIdx][j], if vps_vui_bsp_hrd_parameters( ) syntax structure is present in the active VPS or is available through some external means not specified in this Specification), or equal to sei_bsp_comb_sched_idx[TargetDecLayerSetIdx][SchedSelCombIdx][j] of the bitstream partition HRD parameters SEI message applicable to TargetOp, otherwise.

– The data associated with each decoding unit are removed and decoded instantaneously by the instantaneous decoding process at the CPB removal time of the decoding unit.

– Each decoded picture is placed in the DPB.

– A decoded picture is removed from the DPB when it becomes no longer needed for inter prediction reference and no longer needed for output.

For each bitstream conformance test, the operation of the CPB and the BPB is specified in subclause C.2, the instantaneous decoder operation is specified in clauses 2 through 10, the operation of the DPB is specified in subclause C.3, and the output cropping is specified in subclause C.3.3 and subclause C.5.2.2.

HSS, HBPS and HRD information concerning the number of enumerated delivery schedules and their associated bit rates and buffer sizes is specified in subclauses E.2.2 and E.3.2. The HRD is initialized as specified by the buffering period SEI message specified in subclauses D.2.2 and D.3.2. The removal timing of decoding units from the CPB and output timing of decoded pictures from the DPB is specified using information in picture timing SEI messages (specified in subclauses D.2.3 and D.3.3) or in decoding unit information SEI messages (specified in subclauses D.2.21 and D.3.21). All timing information relating to a specific decoding unit shall arrive prior to the CPB removal time of the decoding unit.

The requirements for bitstream conformance are specified in subclause C.4, and the HRD is used to check conformance of bitstreams as specified above in this subclause and to check conformance of decoders as specified in subclause 11.

NOTE 7 – While conformance is guaranteed under the assumption that all picture-rates and clocks used to generate the bitstream match exactly the values signalled in the bitstream, in a real system each of these may vary from the signalled or specified value.

All the arithmetic in this annex is performed with real values, so that no rounding errors can propagate. For example, the number of bits in a CPB just prior to or after removal of a decoding unit is not necessarily an integer.

The variable ClockTick is derived as follows and is called a clock tick:

$$\text{ClockTick} = \text{vui\_num\_units\_in\_tick} \div \text{vui\_time\_scale} \quad \text{(C-2)}$$

The variable ClockSubTick is derived as follows and is called a clock sub-tick:

$$\text{ClockSubTick} = \text{ClockTick} \div (\text{tick\_divisor\_minus2} + 2) \quad \text{(C-3)}$$

C.2 Operation of coded picture buffer (CPB) and bitstream partition buffer (BPB)

C.2.1 General

The specifications in this subclause apply independently to each set of CPB parameters that is present and to both the Type I and Type II conformance points shown in Figure C-1, and the set of CPB parameters is selected as specified in subclause C.1.

C.2.2 Timing of decoding unit arrival

If SubPicHrdFlag is equal to 0, the variable subPicParamsFlag is set equal to 0, and the process in specified in the remainder of this subclause is invoked with a decoding unit being considered as an access unit, for derivation of the initial and final CPB arrival times for access unit n.

Otherwise (SubPicHrdFlag is equal to 1), the process in specified in the remainder of this subclause is first invoked with the variable subPicParamsFlag set equal to 0 and a decoding unit being considered as an access unit, for derivation of the initial and final CPB arrival times for access unit n, and then invoked with subPicParamsFlag set equal to 1 and a decoding unit being considered as a subset of an access unit, for derivation of the initial and final CPB arrival times for the decoding units in access unit n.

The variables InitCpbRemovalDelay[SchedSelIdx] and InitCpbRemovalDelayOffset[SchedSelIdx] are derived as follows:

– If one or more of the following conditions are true, InitCpbRemovalDelay[SchedSelIdx] and InitCpbRemovalDelayOffset[SchedSelIdx] are set equal to the values of the buffering period SEI message syntax elements nal_initial_alt_cpb_removal_delay[SchedSelIdx] and nal_initial_alt_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 1, or vcl_initial_alt_cpb_removal_delay[SchedSelIdx] and vcl_initial_alt_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 0, where the buffering period SEI message syntax elements are selected as specified in subclause C.1:
  – Access unit 0 is a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_RADL or BLA_N_LP, and the value of irap_cpb_params_present_flag of the buffering period SEI message is equal to 1.
  – Access unit 0 is a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_LP or is a CRA access unit, and the value of irap_cpb_params_present_flag of the buffering period SEI message is equal to 1, and one or more of the following conditions are true:
    – UseAltCpbParamsFlag for access unit 0 is equal to 1.
    – DefaultInitCpbParamsFlag is equal to 0.
  – The value of subPicParamsFlag is equal to 1.
– Otherwise, InitCpbRemovalDelay[SchedSelIdx] and InitCpbRemovalDelayOffset[SchedSelIdx] are set equal to the values of the buffering period SEI message syntax elements nal_initial_cpb_removal_delay[SchedSelIdx] and nal_initial_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 1, or vcl_initial_cpb_removal_delay[SchedSelIdx] and vcl_initial_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 0, where the buffering period SEI message syntax elements are selected as specified in subclause C.1.

The time at which the first bit of decoding unit m begins to enter the CPB is referred to as the initial arrival time initArrivalTime[m].

If the bitstream-specific CPB operation is used, decoding units are indexed in decoding order within the bitstream. Otherwise (the bitstream-partition-specific CPB operation is used), decoding units are indexed in decoding order with each bitstream partition.

The initial arrival time of decoding unit m is derived as follows:

– If the decoding unit is decoding unit 0 (i.e. m = 0) and either the bitstream-specific CPB operation is used or the decoding unit belongs to the base bitstream partition, initArrivalTime[0] = 0.
– Otherwise, if the decoding unit is decoding unit 0, the bitstream-partition-specific CPB operation is used, and the decoding unit does not belong to the base bitstream partition, initArrivalTime[0] is obtained from the bitstream partition initial arrival time SEI message applicable to TargetOp.
– Otherwise, the following applies:
  – If cbr_flag[SchedSelIdx] is equal to 1, the initial arrival time for decoding unit m is equal to the final arrival time (which is derived below) of decoding unit m − 1, i.e.

```
if( !subPicParamsFlag )
    initArrivalTime[ m ] = AuFinalArrivalTime[ m − 1 ]          (C-4)
else
    initArrivalTime[ m ] = DuFinalArrivalTime[ m − 1 ]
```

  – Otherwise (cbr_flag[SchedSelIdx] is equal to 0), the initial arrival time for decoding unit m is derived as follows:

```
if( !subPicParamsFlag )
    initArrivalTime[ m ] = Max( AuFinalArrivalTime[ m − 1 ], initArrivalEarliestTime[ m ] )        (C-5)
else
    initArrivalTime[ m ] = Max( DuFinalArrivalTime[ m − 1 ], initArrivalEarliestTime[ m ] )
```

where initArrivalEarliestTime[m] is derived as follows:

– The variable tmpNominalRemovalTime is derived as follows:

```
if( !subPicParamsFlag )
    tmpNominalRemovalTime = AuNominalRemovalTime[ m ]        (C-6)
else
    tmpNominalRemovalTime = DuNominalRemovalTime[ m ]
```

where AuNominalRemovalTime[m] and DuNominalRemovalTime[m] are the nominal CPB removal time of access unit m and decoding unit m, respectively, as specified in subclause C.2.3.

– If decoding unit m is not the first decoding unit of a subsequent buffering period, initArrivalEarliestTime[m] is derived as follows:

$$\text{initArrivalEarliestTime}[\ m\ ] = \text{tmpNominalRemovalTime} - (\ \text{InitCpbRemovalDelay}[\ \text{SchedSelIdx}\ ] + \text{InitCpbRemovalDelayOffset}[\ \text{SchedSelIdx}\ ]\ ) \div 90000 \quad \text{(C-7)}$$

– Otherwise (decoding unit m is the first decoding unit of a subsequent buffering period), initArrivalEarliestTime[m] is derived as follows:

$$\text{initArrivalEarliestTime}[\ m\ ] = \text{tmpNominalRemovalTime} - (\ \text{InitCpbRemovalDelay}[\ \text{SchedSelIdx}\ ] \div 90000\ ) \quad \text{(C-8)}$$

The final arrival time for decoding unit m is derived as follows:

```
if( !subPicParamsFlag )
    AuFinalArrivalTime[ m ] = initArrivalTime[ m ] + sizeInbits[ m ] ÷ BitRate[ SchedSelIdx ]        (C-9)
else
    DuFinalArrivalTime[ m ] = initArrivalTime[ m ] + sizeInbits[ m ] ÷ BitRate[ SchedSelIdx ]
```

where sizeInbits[m] is the size in bits of decoding unit m, counting the bits of the VCL NAL units and the filler data NAL units for the Type I conformance point or all bits of the Type II bitstream for the Type II conformance point, where the Type I and Type II conformance points are as shown in Figure C-1.

The values of SchedSelIdx, BitRate[SchedSelIdx], and CpbSize[SchedSelIdx] are constrained as follows:

– If the content of the selected hrd_parameters( ) syntax structures for the access unit containing decoding unit m and the previous access unit differ, the HSS selects a value SchedSelIdx1 of SchedSelIdx from among the values of SchedSelIdx provided in the selected hrd_parameters( ) syntax structures for the access unit containing decoding unit m that results in a BitRate[SchedSelIdx1] or CpbSize[SchedSelIdx1] for the access unit containing decoding unit m. The value of BitRate[SchedSelIdx1] or CpbSize[SchedSelIdx1] may differ from the value of BitRate[SchedSelIdx0] or CpbSize[SchedSelIdx0] for the value SchedSelIdx0 of SchedSelIdx that was in use for the previous access unit.

– Otherwise, the HSS continues to operate with the previous values of SchedSelIdx, BitRate[SchedSelIdx] and CpbSize[SchedSelIdx].

When the HSS selects values of BitRate[SchedSelIdx] or CpbSize[SchedSelIdx] that differ from those of the previous access unit, the following applies:

– The variable BitRate[SchedSelIdx] comes into effect at the initial CPB arrival time of the current access unit.

– The variable CpbSize[SchedSelIdx] comes into effect as follows:

  – If the new value of CpbSize[SchedSelIdx] is greater than the old CPB size, it comes into effect at the initial CPB arrival time of the current access unit.

  – Otherwise, the new value of CpbSize[SchedSelIdx] comes into effect at the CPB removal time of the current access unit.

C.2.3 Timing of decoding unit removal and decoding of decoding unit

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

– If one or more of the following conditions are true, CpbDelayOffset is set equal to the value of the buffering period SEI message syntax element cpb_delay_offset, DpbDelayOffset is set equal to the value of the buffering period SEI message syntax element dpb_delay_offset, and InitCpbRemovalDelay[SchedSelIdx] and InitCpbRemovalDelayOffset[SchedSelIdx] are set equal to the values of the buffering period SEI message syntax elements nal_initial_alt_cpb_removal_delay[SchedSelIdx] and nal_initial_alt_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 1, or vcl_initial_alt_cpb_removal_delay[SchedSelIdx] and vcl_initial_alt_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 0, where the buffering period SEI message containing the syntax elements is selected as specified in subclause C.1:

  – Access unit 0 is a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_RADL or BLA_N_LP, and the value of irap_cpb_params_present_flag of the buffering period SEI message is equal to 1.

  – Access unit 0 is a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_LP or is a CRA access unit, and the value of irap_cpb_params_present_flag of the buffering period SEI message is equal to 1, and one or more of the following conditions are true:

    – UseAltCpbParamsFlag for access unit 0 is equal to 1.

    – DefaultInitCpbParamsFlag is equal to 0.

– Otherwise, InitCpbRemovalDelay[SchedSelIdx] and InitCpbRemovalDelayOffset[SchedSelIdx] are set equal to the values of the buffering period SEI message syntax elements nal_initial_cpb_removal_delay[SchedSelIdx] and nal_initial_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 1, or vcl_initial_cpb_removal_delay[SchedSelIdx] and vcl_initial_cpb_removal_offset[SchedSelIdx], respectively, when NalHrdModeFlag is equal to 0, where the buffering period SEI message containing the syntax elements is selected as specified in subclause C.1, CpbDelayOffset and DpbDelayOffset are both set equal to 0.

The nominal removal time of the access unit n from the CPB is specified as follows:

– If access unit n is the access unit with n equal to 0 (the access unit that initializes the HRD), the nominal removal time of the access unit from the CPB is specified by:

AuNominalRemovalTime[0] = InitCpbRemovalDelay[SchedSelIdx] ÷ 90000 (C-10)

– Otherwise, the following applies:

  – When access unit n is the first access unit of a buffering period that does not initialize the HRD, the following applies:

  The nominal removal time of the access unit n from the CPB is specified by:

```
if( !concatenationFlag ) {
    baseTime = AuNominalRemovalTime[ firstPicInPrevBuffPeriod ]
    tmpCpbRemovalDelay = AuCpbRemovalDelayVal
} else {
    baseTime = AuNominalRemovalTime[ prevNonDiscardablePic ]
    tmpCpbRemovalDelay =
        Max( ( auCpbRemovalDelayDeltaMinus1 + 1 ),                                    (C-11)
            Ceil( ( InitCpbRemovalDelay[ SchedSelIdx ] ÷ 90000 +
                AuFinalArrivalTime[ n − 1 ] − AuNominalRemovalTime[ n − 1 ] ) ÷ ClockTick ) )
}
AuNominalRemovalTime[ n ] = baseTime + ClockTick * ( tmpCpbRemovalDelay − CpbDelayOffset )
```

  where AuNominalRemovalTime[firstPicInPrevBuffPeriod] is the nominal removal time of the first access unit of the previous buffering period, 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, AuCpbRemovalDelayVal is the value of AuCpbRemovalDelayVal derived according to au_cpb_removal_delay_minus1 in the picture timing SEI message, selected as specified in subclause C.1, associated with access unit n, and concatenationFlag and auCpbRemovalDelayDeltaMinus1 are the values of the syntax elements concatenation_flag and au_cpb_removal_delay_delta_minus1, respectively, in the buffering period SEI message, selected as specified in subclause C.1, associated with access unit n.

  After the derivation of the nominal CPB removal time and before the derivation of the DPB output time of access unit n, the values of CpbDelayOffset and DpbDelayOffset are updated as follows:

  – If one or more of the following conditions are true, CpbDelayOffset is set equal to the value of the buffering period SEI message syntax element cpb_delay_offset, and DpbDelayOffset is set equal to the value of the buffering period SEI message syntax element dpb_delay_offset, where the buffering period SEI message containing the syntax elements is selected as specified in subclause C.1:

    – Access unit n is a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_RADL or BLA_N_LP, and the value of irap_cpb_params_present_flag of the buffering period SEI message is equal to 1.

    – Access unit n is a BLA access unit for which each coded picture has nal_unit_type equal to BLA_W_LP or is a CRA access unit, and the value of irap_cpb_params_present_flag of the buffering period SEI message is equal to 1, and UseAltCpbParamsFlag for access unit n is equal to 1.

  – Otherwise, CpbDelayOffset and DpbDelayOffset are both set equal to 0.

– When access unit n is not the first access unit of a buffering period, the nominal removal time of the access unit n from the CPB is specified by:

AuNominalRemovalTime[n] = AuNominalRemovalTime[firstPicInCurrBuffPeriod] +
    ClockTick * (AuCpbRemovalDelayVal − CpbDelayOffset) (C-12)

where AuNominalRemovalTime[firstPicInCurrBuffPeriod] is the nominal removal time of the first access unit of the current buffering period, and AuCpbRemovalDelayVal is the value of AuCpbRemovalDelayVal derived according to au_cpb_removal_delay_minus1 in the picture timing SEI message, selected as specified in subclause C.1, associated with access unit n.

When SubPicHrdFlag is equal to 1, the following applies:

– The variable duCpbRemovalDelayInc is derived as follows:

  – If sub_pic_cpb_params_in_pic_timing_sei_flag is equal to 0, duCpbRemovalDelayInc is set equal to the value of du_spt_cpb_removal_delay_increment in the decoding unit information SEI message, selected as specified in subclause C.1, associated with decoding unit m.

  – Otherwise, if du_common_cpb_removal_delay_flag is equal to 0, duCpbRemovalDelayInc is set equal to the value of du_cpb_removal_delay_increment_minus1[i] + 1 for decoding unit m in the picture timing SEI message, selected as specified in subclause C.1, associated with access unit n, where the value of i is 0 for the first num_nalus_in_du_minus1[0] + 1 consecutive NAL units in the access unit that contains decoding unit m, 1 for the subsequent num_nalus_in_du_minus1[1] + 1 NAL units in the same access unit, 2 for the subsequent num_nalus_in_du_minus1[2] + 1 NAL units in the same access unit, etc.

  – Otherwise, duCpbRemovalDelayInc is set equal to the value of du_common_cpb_removal_delay_increment_minus1 + 1 in the picture timing SEI message, selected as specified in subclause C.1, associated with access unit n.

– The nominal removal time of decoding unit m from the CPB is specified as follows, where AuNominalRemovalTime[n] is the nominal removal time of access unit n:

  – If decoding unit m is the last decoding unit in access unit n, the nominal removal time of decoding unit m DuNominalRemovalTime[m] is set equal to AuNominalRemovalTime[n].

  – Otherwise (decoding unit m is not the last decoding unit in access unit n), the nominal removal time of decoding unit m DuNominalRemovalTime[m] is derived as follows:

```
if( sub_pic_cpb_params_in_pic_timing_sei_flag )
    DuNominalRemovalTime[ m ] = DuNominalRemovalTime[ m + 1 ] −
        ClockSubTick * duCpbRemovalDelayInc                                   (C-13)
else
    DuNominalRemovalTime[ m ] = AuNominalRemovalTime[ n ] −
        ClockSubTick * duCpbRemovalDelayInc
```

If SubPicHrdFlag is equal to 0, the removal time of access unit n from the CPB is specified as follows, where AuFinalArrivalTime[n] and AuNominalRemovalTime[n] are the final CPB arrival time and nominal CPB removal time, respectively, of access unit n:

```
if( !low_delay_hrd_flag[ HighestTid ] || AuNominalRemovalTime[ n ] >= AuFinalArrivalTime[ n ] )
    AuCpbRemovalTime[ n ] = AuNominalRemovalTime[ n ]
else                                                                          (C-14)
    AuCpbRemovalTime[ n ] = AuNominalRemovalTime[ n ] + ClockTick *
        Ceil( ( AuFinalArrivalTime[ n ] − AuNominalRemovalTime[ n ] ) ÷ ClockTick )
```

NOTE 1 – When low_delay_hrd_flag[HighestTid] is equal to 1 and AuNominalRemovalTime[n] is less than AuFinalArrivalTime[n], the size of access unit n is so large that it prevents removal at the nominal removal time.

Otherwise (SubPicHrdFlag is equal to 1), the removal time of decoding unit m from the CPB is specified as follows:

– When the bitstream-specific CPB operation is used or when the current DU belongs to the base bitstream partition, the following applies:

```
if( !low_delay_hrd_flag[ HighestTid ] || DuNominalRemovalTime[ m ] >= DuFinalArrivalTime[ m ] )
    DuCpbRemovalTime[ m ] = DuNominalRemovalTime[ m ]
else                                                                          (C-15)
    DuCpbRemovalTime[ m ] = DuFinalArrivalTime[ m ]
```

NOTE 2 – When low_delay_hrd_flag[HighestTid] is equal to 1 and DuNominalRemovalTime[m] is less than DuFinalArrivalTime[m], the size of decoding unit m is so large that it prevents removal at the nominal removal time.

– When the bitstream-partition-specific CPB operation is used and cbr_flag[SchedSelIdx] is equal to 0, the following applies:

  – Let refDuCpbRemovalTime be equal to the CPB removal time of the previous DU preceding the current DU in decoding order (regardless of the bitstream partitions to which the previous DU and the current DU belong).

  – The variable DuCpbRemovalTime[m] is modified as follows:

DuCpbRemovalTime[m] = Max(DuCpbRemovalTime[m], refDuCpbRemovalTime) (C-16)

If SubPicHrdFlag is equal to 0, at the CPB removal time of access unit n, the access unit is instantaneously decoded.

Otherwise (SubPicHrdFlag is equal to 1), at the CPB removal time of decoding unit m, the decoding unit is instantaneously decoded, and when decoding unit m is the last decoding unit of access unit n, the following applies:

– Access unit n is considered as decoded.

– The final CPB arrival time of access unit n, i.e. AuFinalArrivalTime[n], is set equal to the final CPB arrival time of the last decoding unit in access unit n, i.e. DuFinalArrivalTime[m].

– The nominal CPB removal time of access unit n, i.e. AuNominalRemovalTime[n], is set equal to the nominal CPB removal time of the last decoding unit in access unit n, i.e. DuNominalRemovalTime[m].

– The CPB removal time of access unit n, i.e. AuCpbRemovalTime[m], is set equal to the CPB removal time of the last decoding unit in access unit n, i.e. DuCpbRemovalTime[m].

C.3 Operation of the decoded picture buffer (DPB)

C.3.1 General

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

The decoded picture buffer consists of sub-DPBs, and each sub-DPB contains picture storage buffers for storage of decoded pictures of one layer only. Each of the picture storage buffers of a sub-DPB may contain a decoded picture that is marked as "used for reference" or is held for future output.

The following applies for all decoded access units:

– If an access unit does not contain a picture at a target output layer and alt_output_layer_flag is equal to 1, the following ordered steps apply:

  – The list nonOutputLayerPictures is the list of pictures of the access unit with PicOutputFlag equal to 1 and with nuh_layer_id values that are included in the TargetDecLayerIdList and that are not on target output layers.

  – The picture with the highest nuh_layer_id value among the list nonOutputLayerPictures is removed from the list nonOutputLayerPictures.

  – PicOutputFlag for each picture that is included in the list nonOutputLayerPictures is set equal to 0.

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

The processes specified in subclauses C.3.2, C.3.3 and C.3.4 are sequentially applied as specified below, and are applied independently for each layer, starting from the base layer, in increasing order of nuh_layer_id values of the layers in the bitstream. When these processes are applied for a particular layer, only the sub-DPB for the particular layer is affected. In the descriptions of these processes, the DPB refers to the sub-DPB for the particular layer, and the particular layer is referred to as the current layer.

NOTE – In the operation of output timing DPB, decoded pictures with PicOutputFlag equal to 1 in the same access unit are output consecutively in ascending order of the nuh_layer_id values of the decoded pictures.

Let picture n and the current picture 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.]

C.3.2 Removal of pictures from the DPB

When the current picture is not picture 0 in the current layer, the removal of pictures in the current layer from the DPB before decoding of the current picture, i.e. picture n, 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 current picture and proceeds as follows:

– The decoding process for RPS as specified in subclause 8.3.2 is invoked.

– When the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, or the base layer picture in the current access unit is an IRAP picture with NoRaslOutputFlag equal to 1 and NoClrasOutputFlag is equal to 1, 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 with NoRaslOutputFlag equal to 1, NoOutputOfPriorPicsFlag is set equal to 1 (regardless of the value of no_output_of_prior_pics_flag).

   – Otherwise, if the current picture is an IRAP picture with NoRaslOutputFlag equal to 1 and 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 for the current layer 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 that was active for the current layer when decoding the preceding picture in the current layer, 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 – 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, if the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, NoOutputOfPriorPicsFlag is set equal to no_output_of_prior_pics_flag.

   – Otherwise (the current picture is not an IRAP picture with NoRaslOutputFlag equal to 1, the base layer picture in the current access unit is an IRAP picture with NoRaslOutputFlag equal to 1, and NoClrasOutputFlag is equal to 1), NoOutputOfPriorPicsFlag is set equal to 1.

2. The value of NoOutputOfPriorPicsFlag derived for the decoder under test is applied for the HRD, such that when the value of 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.

– When both of the following conditions are true for any pictures k in the DPB, all such pictures k in the DPB are removed from the DPB:

  – picture k is marked as "unused for reference"

  – picture k has PicOutputFlag equal to 0 or its DPB output time is less than or equal to the CPB removal time of the first decoding unit (denoted as decoding unit m) of the current picture n; i.e. DpbOutputTime[k] is less than or equal to CpbRemovalTime(m)

– For each picture that is removed from the DPB, the DPB fullness is decremented by one.

C.3.3 Picture output

The processes specified in this subclause happen instantaneously at the CPB removal time of access unit n, AuCpbRemovalTime[n].

When picture n has PicOutputFlag equal to 1, its DPB output time DpbOutputTime[n] is derived as follows, where the variable firstPicInBufferingPeriodFlag is equal to 1 if access unit n is the first access unit of a buffering period and 0 otherwise:

```
if( !SubPicHrdFlag ) {
    DpbOutputTime[ n ] = AuCpbRemovalTime[ n ] + ClockTick * picDpbOutputDelay          (C-17)
    if( firstPicInBufferingPeriodFlag )
        DpbOutputTime[ n ] −= ClockTick * DpbDelayOffset
```

} else
DpbOutputTime[n] = AuCpbRemovalTime[n] + ClockSubTick * picSptDpbOutputDuDelay

where picDpbOutputDelay is the value of pic_dpb_output_delay in the picture timing SEI message associated with access unit n, and picSptDpbOutputDuDelay is the value of pic_spt_dpb_output_du_delay, when present, in the decoding unit information SEI messages associated with access unit n, or the value of pic_dpb_output_du_delay in the picture timing SEI message associated with access unit n when there is no decoding unit information SEI message associated with access unit n or no decoding unit information SEI message associated with access unit n has pic_spt_dpb_output_du_delay present.

NOTE – When the syntax element pic_spt_dpb_output_du_delay is not present in any decoding unit information SEI message associated with access unit n, the value is inferred to be equal to pic_dpb_output_du_delay in the picture timing SEI message associated with access unit n.

The output of the current picture is specified as follows:

- If PicOutputFlag is equal to 1 and DpbOutputTime[n] is equal to AuCpbRemovalTime[n], the current picture is output.
- Otherwise, if PicOutputFlag is equal to 0, the current picture is not output, but will be stored in the DPB as specified in subclause C.3.4.
- Otherwise (PicOutputFlag is equal to 1 and DpbOutputTime[n] is greater than AuCpbRemovalTime[n]), the current picture is output later and will be stored in the DPB (as specified in subclause C.3.4) and is output at time DpbOutputTime[n] unless indicated not to be output by the decoding or inference of no_output_of_prior_pics_flag equal to 1 at a time that precedes DpbOutputTime[n].

When output, the picture is cropped, using the conformance cropping window specified in the active SPS for the layer containing the picture.

When picture n is a picture that is output and is not the last picture of the bitstream that is output, the value of the variable DpbOutputInterval[n] is derived as follows:

DpbOutputInterval[n] = DpbOutputTime[nextPicInOutputOrder] − DpbOutputTime[n] (C-18)

where nextPicInOutputOrder is the picture that follows picture n in output order and has PicOutputFlag equal to 1.

C.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 the current picture. [Ed. (MH): This change might not comply with version 1, because version 1 decoders would mark and store the base-layer picture at the CPB removal time of the AU, which can be later than the CPB removal time of the base-layer picture.]

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".

C.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.

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

Let currPicLayerId be equal to the nuh_layer_id of the current picture.

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]) (C-19)

   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]) (C-20)

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

   Floor(deltaTime90k[n]) <= InitCpbRemovalDelay[SchedSelIdx] <= Ceil(deltaTime90k[n]) (C-21)

   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. When the bitstream-partition-specific CPB operation is used and cbr_flag[SchedSelIdx] is equal to 1, DuCpbRemovalTime[m] shall be greater than or equal to the CPB removal time of the previous DU preceding the current DU in decoding order (regardless of the bitstream partitions to which the previous DU and the current DU belong) for any decoding unit m in bitstream partitions with index greater than 0.

6. 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.

7. For each current picture, after invocation of the process for removal of pictures from the sub-DPB as specified in subclause C.3.2, the number of decoded pictures in the sub-DPB for the current layer, including all pictures n in the current layer 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] when currPicLayerId is equal to 0 or max_vps_dec_pic_buffering_minus1[TargetOptLayerSetIdx][currPicLayerId][HighestTid] when currPicLayerId is greater than 0.

8. 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.

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

10. The value of DpbOutputInterval[n] as given by Equation C-18, which is the difference between the output time of an access unit 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).]

11. 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++ )                                  (C-22)
    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.

12. For any two pictures m and n in the same CVS, when DpbOutputTime[m] is greater than DpbOutputTime[n], the PicOrderCntVal of picture m shall be greater than the PicOrderCntVal of picture n.

NOTE 2 – All pictures of an earlier CVS in decoding order that are output are output before any pictures of a later CVS in decoding order. Within any particular CVS, the pictures that are output are output in increasing PicOrderCntVal order.

C.5 Decoder conformance

C.5.1 General

A decoder conforming to this Specification shall fulfil all requirements specified in this subclause.

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

When a bitstream contains syntax elements that have values that are specified as reserved and it is specified that decoders shall ignore values of the syntax elements or NAL units containing the syntax elements having the reserved values, and the bitstream is otherwise conforming to this Specification, a conforming decoder shall decode the bitstream in the same manner as it would decode a conforming bitstream and shall ignore the syntax elements or the NAL units containing the syntax elements having the reserved values as specified.

There are two types of conformance that can be claimed by a decoder: output timing conformance and output order conformance.

To check conformance of a decoder, test bitstreams conforming to the claimed profile, tier and level, as specified in subclause C.4 are delivered by a hypothetical stream scheduler (HSS) both to the HRD and to the decoder under test (DUT). All cropped decoded pictures output by the HRD shall also be output by the DUT, each cropped decoded picture output by the DUT shall be a picture with PicOutputFlag equal to 1, and, for each such cropped decoded picture output by the DUT, the values of all samples that are output shall be equal to the values of the samples produced by the specified decoding process.

For output timing decoder conformance, the HSS operates as described above, with delivery schedules selected only from the subset of values of SchedSelIdx for which the bit rate and CPB size are restricted as specified in Annex A for the specified profile, tier and level, or with "interpolated" delivery schedules as specified below for which the bit rate and CPB size are restricted as specified in Annex A. The same delivery schedule is used for both the HRD and the DUT.

When the HRD parameters and the buffering period SEI messages are present with cpb_cnt_minus1[HighestTid] greater than 0, the decoder shall be capable of decoding the bitstream as delivered from the HSS operating using an "interpolated" delivery schedule specified as having peak bit rate r, CPB size c(r), and initial CPB removal delay (f(r) ÷ r) as follows:

$$\alpha = (r - \text{BitRate}[\text{SchedSelIdx} - 1]) \div (\text{BitRate}[\text{SchedSelIdx}] - \text{BitRate}[\text{SchedSelIdx} - 1]), \tag{C-23}$$

$$c(r) = \alpha * \text{CpbSize}[\text{SchedSelIdx}] + (1 - \alpha) * \text{CpbSize}[\text{SchedSelIdx} - 1], \tag{C-24}$$

$$f(r) = \alpha * \text{InitCpbRemovalDelay}[\text{SchedSelIdx}] * \text{BitRate}[\text{SchedSelIdx}] + (1 - \alpha) * \text{InitCpbRemovalDelay}[\text{SchedSelIdx} - 1] * \text{BitRate}[\text{SchedSelIdx} - 1] \tag{C-25}$$

for any SchedSelIdx > 0 and r such that BitRate[SchedSelIdx − 1] <= r <= BitRate[SchedSelIdx] such that r and c(r) are within the limits as specified in Annex A for the maximum bit rate and buffer size for the specified profile, tier and level.

NOTE 1 – InitCpbRemovalDelay[SchedSelIdx] can be different from one buffering period to another and have to be re-calculated.

For output timing decoder conformance, an HRD as described above is used and the timing (relative to the delivery time of the first bit) of picture output is the same for both the HRD and the DUT up to a fixed delay.

For output order decoder conformance, the following applies:

– The HSS delivers the bitstream BitstreamToDecode to the DUT "by demand" from the DUT, meaning that the HSS delivers bits (in decoding order) only when the DUT requires more bits to proceed with its processing.

  NOTE 2 – This means that for this test, the coded picture buffer of the DUT could be as small as the size of the largest decoding unit.

– A modified HRD as described below is used, and the HSS delivers the bitstream to the HRD by one of the schedules specified in the bitstream BitstreamToDecode such that the bit rate and CPB size are restricted as specified in Annex A. The order of pictures output shall be the same for both the HRD and the DUT.

– The HRD CPB size is given by CpbSize[SchedSelIdx] as specified in subclause E.3.3, where SchedSelIdx and the HRD parameters are selected as specified in subclause C.1. The DPB size is given by sps_max_dec_pic_buffering_minus1[HighestTid] + 1. Removal time from the CPB for the HRD is the final bit arrival time and decoding is immediate. The operation of the DPB of this HRD is as described in subclauses C.5.2 through C.5.2.3.

C.5.2 Operation of the output order DPB

C.5.2.1 General

The decoded picture buffer consists of sub-DPBs, and each sub-DPB contains picture storage buffers for storage of decoded pictures of one layer only. Each of the picture storage buffers of a sub-DPB 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 C.5.2.2 is invoked, followed by the invocation of the process for picture decoding, marking, additional bumping, and storage as specified in subclause C.5.2.3. The "bumping" process is specified in subclause C.5.2.4 and is invoked as specified in subclauses C.5.2.2 and C.5.2.3.

These processes are applied independently for each layer, starting from the base layer, in increasing order of the nuh_layer_id values of the layers in the bitstream. When these processes are applied for a particular layer, only the sub-DPB for the particular layer is affected except for the "bumping" process, which may crop and output pictures, mark pictures as "not needed for output" and empty picture storage buffers for any layer.

NOTE – In the operation of output order DPB, same as in the operation of output timing DPB, decoded pictures with PicOutputFlag equal to 1 in the same access unit are also output consecutively in ascending order of the nuh_layer_id values of the decoded pictures.

Let picture n and the current picture 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.

When these processes are applied for a layer with nuh_layer_id equal to currLayerId, the variables MaxNumReorderPics, MaxLatencyIncreasePlus1, MaxLatencyPictures, and MaxDecPicBufferingMinus1 are derived as follows:

– If a CVS conforming to one or more of the profiles specified in Annex G or H is decoded by applying the decoding process specified in clauses 2−10, Annex F, and Annex G or H, the following applies:

  – MaxNumReorderPics is set equal to max_vps_num_reorder_pics[TargetOptLayerSetIdx][HighestTid] of the active VPS.

  – MaxLatencyIncreasePlus1 is set equal to the value of the syntax element max_vps_latency_increase_plus1[TargetOptLayerSetIdx][HighestTid] of the active VPS,.

  – MaxLatencyPictures is set equal to VpsMaxLatencyPictures[TargetOptLayerSetIdx][HighestTid] of the active VPS.

  – MaxDecPicBufferingMinus1 is set equal to the value of the syntax element max_vps_dec_pic_buffering_minus1[TargetOptLayerSetIdx][currLayerId][HighestTid] of the active VPS.

– Otherwise (a CVS conforming to one or more of the profiles specified in Annex A is decoded by applying the decoding process specified in clauses 2−10), the following applies:

  – MaxNumReorderPics is set equal to sps_max_num_reorder_pics[HighestTid] of the active SPS for the base layer.

  – MaxLatencyIncreasePlus1 is set equal to sps_max_latency_increase_plus1[HighestTid] of the active SPS for the base layer.

– MaxLatencyPictures is set equal to SpsMaxLatencyPictures[HighestTid] of the active SPS for the base layer.

– MaxDecPicBufferingMinus1 is set equal to sps_max_dec_pic_buffering_minus1[HighestTid] of the active SPS for the base layer.

C.5.2.2 Output and removal of pictures from the DPB

When the current picture is not picture 0 in the current layer, the output and removal of pictures in the current layer from the DPB before the decoding of the current picture , i.e. picture n, 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.

– If the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, or the base layer picture in the current access unit is an IRAP picture with NoRaslOutputFlag equal to 1 and NoClrasOutputFlag is equal to 1, 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 with NoRaslOutputFlag equal to 1, NoOutputOfPriorPicsFlag is set equal to 1 (regardless of the value of no_output_of_prior_pics_flag).

   – Otherwise, if the current picture is an IRAP picture with NoRaslOutputFlag equal to 1 and 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 for the current layer 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 that was active for the current layer when decoding the preceding picture in the current layer, 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 – 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, if the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, NoOutputOfPriorPicsFlag is set equal to no_output_of_prior_pics_flag.

   – Otherwise (the current picture is not an IRAP picture with NoRaslOutputFlag equal to 1, the base layer picture in the current access unit is an IRAP picture with NoRaslOutputFlag equal to 1, and NoClrasOutputFlag is equal to 1), NoOutputOfPriorPicsFlag is set equal to 1.

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 sub-DPB are emptied without output of the pictures they contain, and the sub-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 sub-DPB are emptied by repeatedly invoking the "bumping" process specified in subclause C.5.2.4, and the sub-DPB fullness is set equal to 0.

– Otherwise, all picture storage buffers that contain a picture in the current layer and that are marked as "not needed for output" and "unused for reference" are emptied (without output). For each picture storage buffer that is emptied, the sub-DPB fullness is decremented by one. When one or more of the following conditions are true, the "bumping" process specified in subclause C.5.2.4 is invoked repeatedly while further decrementing the sub-DPB fullness by one for each additional picture storage buffer that is emptied, until none of the following conditions are true:

  – The number of access units that contain at least one decoded picture in the DPB marked as "needed for output" is greater than MaxNumReorderPics.

  – MaxLatencyIncreasePlus1 is not equal to 0 and there is at least one access units that contain at least one decoded picture in the DPB marked as "needed for output" for which the associated variable PicLatencyCount is greater than or equal to MaxLatencyPictures.

  – The number of pictures in the current layer in the sub-DPB is greater than or equal to MaxDecPicBufferingMinus1 + 1.

C.5.2.3 Picture decoding, marking, additional bumping, and storage

The processes specified in this subclause happen instantaneously when the last decoding unit of picture n is removed from the CPB. [Ed. (MH): This change might not comply with version 1, because version 1 decoders would mark and store the base-layer picture at the CPB removal time of the AU, which can be later than the CPB removal time of the base-layer picture.]

PicOutputFlag is updated as follows:

– If the current access unit does not contain a picture at a target output layer and alt_output_layer_flag is equal to 1, the following ordered steps apply:
  – The list nonOutputLayerPictures is the list of pictures of the access unit with PicOutputFlag equal to 1 and with nuh_layer_id values that are included in the TargetDecLayerIdList and that are not on target output layers.
  – The picture with the highest nuh_layer_id value among the list nonOutputLayerPictures is removed from the list nonOutputLayerPictures.
  – PicOutputFlag for each picture that is included in the list nonOutputLayerPictures is set equal to 0.
– Otherwise, PicOutputFlag for pictures that are not included in a target output layer is set equal to 0.

When the current picture has PicOutputFlag equal to 1, for each picture in the current layer in the sub-DPB that is marked as "needed for output" and follows the current picture in output order, the associated variable PicLatencyCount is set equal to PicLatencyCount + 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 sub-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 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 C.5.2.4 is invoked repeatedly until none of the following conditions are true:

– The number of access units that contain at least one decoded picture in the DPB marked as "needed for output" is greater than MaxNumReorderPics.
– MaxLatencyIncreasePlus1 is not equal to 0 and there is at least one access units that contain at least one decoded picture in the DPB marked as "needed for output" for which the associated variable PicLatencyCount is greater than or equal to MaxLatencyPictures.

C.5.2.4 "Bumping" process

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

1. The picture or 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. Each of these pictures is, in ascending nuh_layer_id order, cropped, using the conformance cropping window specified in the active SPS for the picture, the cropped picture is output, and the picture is marked as "not needed for output".
3. Each picture storage buffer that contains a picture marked as "unused for reference" and that was one of the pictures cropped and output is emptied.

C.6 Demultiplexing process for deriving a bitstream partition

Inputs to this process are a bitstream, a layer identifier list bspLayerId[bspIdx] and the number of layer identifiers numBspLayerId in the layer index list bspLayerId[bspIdx].

Output of this process is a bitstream partition.

Let variable minBspLayerId be the smallest value of bspLayerId[bspIdx] with any value of bspIdx in the range of 0 to numBspLayerId – 1, inclusive.

The output bitstream partition consists of selected NAL units of the input bitstream in the same order as they appear in the input bitstream. The following NAL units of the input bitstream are omitted from the output bitstream partition, while the remaining NAL units of the input bitstream are included in the output bitstream partition:

– Omit all NAL units that have a nuh_layer_id value other than bspLayerId[bspIdx] with any value of bspIdx in the range of 0 to numBspLayerId – 1, inclusive.

– Omit all SEI NAL units containing a scalable nesting SEI message for which no derived nestingLayerIdList[i] contains any layer identifier value equal to bspLayerId[bspIdx] with any value of bspIdx in the range of 0 to numBspLayerId – 1, inclusive.

– Omit all SEI NAL units containing a scalable nesting SEI message for which a derived nestingLayerIdList[i] contains a layer identifier value less than minBspLayerId.

Modify subclause D.1.1 as follows:

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

Add rows enclosed by "...".

sei_payload(payloadType, payloadSize) {	**Descriptor**
if(nal_unit_type = = PREFIX_SEI_NUT)	
if(payloadType = = 0)	
...	
else if(payloadType = = XXX)	
layers_not_present(payloadSize)	
else if(payloadType = = XXX)	
inter_layer_constrained_tile_sets(payloadSize)	
else if(payloadType = = XXX)	
bsp_nesting(payloadSize)	
else if(payloadType = = XXX)	
bsp_initial_arrival_time(payloadSize)	
else if(payloadType = = XXX)	
bsp_hrd(payloadSize)	
else if(payloadType = = XXX)	
three_dimensional_reference_displays_info(payloadSize)	
else if(payloadType = = XXX)	
depth_representation_info_sei(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)
}	
}	

Modify subclause E.2.1 as follows:

Annex E

Video usability information

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

E.2 VUI semantics

E.2.1 VUI parameters semantics

The specifications in clause E.2.1 apply with the following modifications and additions.

video_signal_type_present_flag equal to 1 specifies that video_format, video_full_range_flag and colour_description_present_flag are present. video_signal_type_present_flag equal to 0, specifies that video_format, video_full_range_flag and colour_description_present_flag are not present. It is a requirement of bitstream conformance that, when nuh_layer_id is greater than 0, video_signal_type_present_flag shall be equal to 0.

When a current picture with nuh_layer_id layerIdCurr greater than 0 refers to an SPS containing the VUI parameter syntax structure, the values of video_format, video_full_range_flag, colour_primaries, transfer_characteristics, and matrix_coeffs are inferred as follows:

– If the nuh_layer_id of the active SPS for the layer with nuh_layer_id equal to layerIdCurr is equal to 0, the values of video_format, video_full_range_flag, colour_primaries, transfer_characteristics, and matrix_coeffs are inferred to be equal to video_vps_format, video_full_range_vps_flag, colour_primaries_vps, transfer_characteristics_vps, and matrix_coeffs_vps, respectively, of the vps_video_signal_info_idx[j]-th video_signal_info( ) syntax structure in the active VPS where j is equal to LayerIdxInVps[layerIdCurr] and the values of video_format, video_full_range_flag, colour_primaries, transfer_characteristics, and matrix_coeffs of the active SPS for the layer with nuh_layer_id equal to layerIdCurr are ignored.

  NOTE X – The values are inferred from the VPS when a non-base layer refers to an SPS that is also referred to by the base layer, in which case the SPS has nuh_layer_id equal to 0. For the base layer, the values of these parameters in the active SPS for the base layer apply.

– Otherwise (the nuh_layer_id of the active SPS for the layer with nuh_layer_id equal to layerIdCurr is greater than zero), values of video_format, video_full_range_flag, colour_primaries, transfer_characteristics, and matrix_coeffs are inferred to be equal to video_vps_format, video_full_range_vps_flag, colour_primaries_vps, transfer_characteristics_vps, and matrix_coeffs_vps, respectively, of the vps_video_signal_info_idx[j]-th video_signal_info( ) syntax structure in the active VPS, where j is equal to LayerIdxInVps[layerIdCurr].

[Ed. (GT) Consider shortening duplicated inference specification above. What should happen when VPS VUI is not present?]

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, output order, picture order count, RADL AU, RASL AU, (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.

NOTE 1 – 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 **auxiliary picture**: A *picture* that has no normative effect on the *decoding process* of *primary pictures*.

F.3.4 **base layer:** A *layer* in which all *VCL NAL units* have nuh_layer_id equal to 0.

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 **coded video sequence (CVS):** A sequence of *access units* that consists, in decoding order, of an *initial* IRAP *access unit*, followed by zero or more *access units* that are not *initial* IRAP *access units*, including all subsequent *access units* up to but not including any subsequent *access unit* that is an *initial* IRAP *access unit*.

F.3.7 **collocated sample:** A sample TBD. [Ed. (GT) Maybe it is easier to define a collocated position and require collocated samples to have it?]

F.3.8 **direct reference layer:** A *layer* that may be used for inter-layer prediction of another *layer*.

F.3.9 **indirect reference layer:** A *layer* that is not a *direct reference layer* of another *layer* but is a *direct reference layer* of a *layer* that is a *direct reference layer* or indirect reference *layer* of a *direct reference layer* of the *layer*.

F.3.10 **initial intra random access point (IRAP) access unit**: An *IRAP access unit* in which the *coded picture* with nuh_layer_id equal to 0 has NoRaslOutputFlag equal to 1.

F.3.11 **inter-layer prediction:** A *prediction* in a manner that is dependent on data elements (e.g. sample values or motion vectors) of *reference pictures* with a different value of nuh_layer_id than that of the current *picture*.

F.3.12 **intra random access point (IRAP) access unit**: An *access unit* in which the *coded picture* with nuh_layer_id equal to 0 is an *IRAP picture*.

F.3.13 **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.3.14 **non-base layer:** A *layer* in which all *VCL NAL units* have the same nuh_layer_id value greater than 0.

F.3.15 **picture order count:** A variable that is associated with each *picture* and that uniquely identifies the associated *picture* among all *pictures* with the same 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 the same value of nuh_layer_id in the same *CVS* that are to be output from the *decoded picture buffer*.

F.3.16 **primary picture**: a *picture* with a nuh_layer_id value such that AuxId[nuh_layer_id] is equal to 0.

F.3.17 **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.3.18 **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.3.19 **target output layer:** A *layer* that is to be output.

F.3.20 **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.3.21 **output time:** A time when a *decoded picture* is to be output as specified in Annex C, if the timing information is present in the *coded video sequence*. [Ed.: Consider adding this definition in clause 3 of the main specification containing both version 1 and Annex F specifications.]

F.3.22 **view:** A sequence of pictures associated with the same value of ViewOrderIdx.

NOTE 2 – A view typically represents a sequence of pictures captured by one camera.

F.4 Abbreviations

The specifications in clause 4 apply.

F.5 Conventions

The specifications in clause 5 apply.

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

The specifications 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) {	
while(!byte_aligned( ))	
vps_extension_alignment_bit_equal_to_one	u(1)

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
avc_base_layer_flag	u(1)
vps_vui_present_flag	u(1)
if(vps_vui_present_flag)	
vps_vui_offset	u(16)
splitting_flag	u(1)
for(i = 0, NumScalabilityTypes = 0; i < 16; i++) {	
scalability_mask_flag[i]	u(1)
NumScalabilityTypes += scalability_mask_flag[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 = 1; i <= MaxLayersMinus1; i++) {	
if(vps_nuh_layer_id_present_flag)	
layer_id_in_nuh[i]	u(6)
if(!splitting_flag)	
for(j = 0; j < NumScalabilityTypes; j++)	
dimension_id[i][j]	u(v)
}	
view_id_len	u(4)
if(view_id_len > 0)	
for(i = 0; i < NumViews; i++)	
view_id_val[i]	u(v)
for(i = 1; i <= MaxLayersMinus1; i++)	
for(j = 0; j < i; j++)	
direct_dependency_flag[i][j]	u(1)
vps_sub_layers_max_minus1_present_flag	u(1)
if(vps_sub_layers_ max_minus1_present_flag)	
for(i = 0; i <= MaxLayersMinus1; i++)	
sub_layers_vps_max_minus1[i]	u(3)
max_tid_ref_present_flag	u(1)
if(max_tid_ref_present_flag)	
for(i = 0; i < MaxLayersMinus1; i++)	
for(j = i + 1; j <= MaxLayersMinus1; j++)	
if(direct_dependency_flag[j][i])	
max_tid_il_ref_pics_plus1[i][j]	u(3)
all_ref_layers_active_flag	u(1)
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_idc	u(2)
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)
}	
if(vps_max_layers_minus1 > 0)	
alt_output_layer_flag	u(1)
rep_format_idx_present_flag	u(1)
if(rep_format_idx_present_flag)	
vps_num_rep_formats_minus1	u(4)
for(i = 0; i <= vps_num_rep_formats_minus1; i++)	
rep_format( )	
if(rep_format_idx_present_flag)	
for(i = 1; i <= MaxLayersMinus1; i++)	
if(vps_num_rep_formats_minus1 > 0)	
vps_rep_format_idx[i]	u(8)
max_one_active_ref_layer_flag	u(1)
for(i = 1; i <= MaxLayersMinus1; i++)	
if(NumDirectRefLayers[layer_id_in_nuh[i]] = = 0)	
poc_lsb_not_present_flag[i]	u(1)
dpb_size( )	
direct_dep_type_len_minus2	ue(v) [Ed. (JB): Should this be ue(v)?]
default_direct_dependency_flag	u(1)
if(default_direct_dependency_flag)	
default_direct_dependency_type	u(v)
else {	
for(i = 1; i <= MaxLayersMinus1; 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)
if(vps_vui_present_flag) {	
while(!byte_aligned( ))	

vps_vui_alignment_bit_equal_to_one	u(1)
vps_vui( )	
}	
}	

F.7.3.2.1.2 Representation format syntax

[Ed. (YK): The syntax and semantics for rep_format( ), dpb_size( ), and vps_vui( ) should probably have one-level-higher section titles, similarly as profile_tier_level( ).]

rep_format( ) {	Descriptor
pic_width_vps_in_luma_samples	u(16)
pic_height_vps_in_luma_samples	u(16)
chroma_and_bit_depth_vps_present_flag	u(1)
if(chroma_and_bit_depth_vps_present_flag) {	
chroma_format_vps_idc	u(2)
if(chroma_format_vps_idc = = 3)	
separate_colour_plane_vps_flag	u(1)
bit_depth_vps_luma_minus8	u(4)
bit_depth_vps_chroma_minus8	u(4)
}	
}	

F.7.3.2.1.3 DPB size syntax

dpb_size( ) {	
for(i = 1; i < NumOutputLayerSets; i++) {	
sub_layer_flag_info_present_flag[i]	u(1)
for(j = 0; j <= vps_max_sub_layers_minus1; j++) {	
if(j > 0 && sub_layer_flag_info_present_flag[i])	
sub_layer_dpb_info_present_flag[i][j]	u(1)
if(sub_layer_dpb_info_present_flag[i][j]) {	
for(k = 0; k < NumSubDpbs[i]; k++)	
max_vps_dec_pic_buffering_minus1[i][k][j]	ue(v)
max_vps_num_reorder_pics[i][j]	ue(v)
max_vps_latency_increase_plus1[i][j]	ue(v)
}	
}	
}	
}	

F.7.3.2.1.4 VPS VUI syntax

vps_vui( ){	Descriptor
cross_layer_pic_type_aligned_flag	u(1)
if(!cross_layer_pic_type_aligned_flag)	
cross_layer_irap_aligned_flag	u(1)
bit_rate_present_vps_flag	u(1)
pic_rate_present_vps_flag	u(1)
if(bit_rate_present_vps_flag || pic_rate_present_vps_flag)	
for(i = 0; i <= vps_number_layer_sets_minus1; i++)	
for(j = 0; j <= vps_max_sub_layers_minus1; j++) {	
if(bit_rate_present_vps_flag)	
bit_rate_present_flag[i][j]	u(1)
if(pic_rate_present_vps_flag)	
pic_rate_present_flag[i][j]	u(1)
if(bit_rate_present_flag[i][j]) {	
avg_bit_rate[i][j]	u(16)
max_bit_rate[i][j]	u(16)
}	
if(pic_rate_present_flag[i][j]) {	
constant_pic_rate_idc[i][j]	u(2)
avg_pic_rate[i][j]	u(16)
}	
}	
tiles_not_in_use_flag	u(1)
if(!tiles_not_in_use_flag) {	
for(i = 0; i <= MaxLayersMinus1; i++) {	u(1)
tiles_in_use_flag[i]	
if(tiles_in_use_flag[i])	
loop_filter_not_across_tiles_flag[i]	u(1)
}	
for(i = 1; i <= MaxLayersMinus1; i++)	
for(j = 0; j < NumDirectRefLayers[layer_id_in_nuh[i]]; j++) {	
layerIdx = LayerIdxInVps[RefLayerId[layer_id_in_nuh[i]][j]]	
if(tiles_in_use_flag[i] && tiles_in_use_flag[layerIdx])	
tile_boundaries_aligned_flag[i][j]	u(1)
}	
}	
wpp_not_in_use_flag	
if(!wpp_not_in_use_flag)	
for(i = 0; i <= MaxLayersMinus1; i++)	
wpp_in_use_flag[i]	u(1)
ilp_restricted_ref_layers_flag	u(1)
if(ilp_restricted_ref_layers_flag)	
for(i = 1; i <= MaxLayersMinus1; i++)	
for(j = 0; j < NumDirectRefLayers[layer_id_in_nuh[i]]; j++) {	
min_spatial_segment_offset_plus1[i][j]	ue(v)
if(min_spatial_segment_offset_plus1[i][j] > 0) {	

ctu_based_offset_enabled_flag[i][j]	u(1)
if(ctu_based_offset_enabled_flag[i][j])	
min_horizontal_ctu_offset_plus1[i][j]	ue(v)
}	
}	
video_signal_info_idx_present_flag	u(1)
if(video_signal_info_idx_present_flag)	
vps_num_video_signal_info_minus1	u(4)
for(i = 0; i <= vps_num_video_signal_info_minus1; i++)	
video_signal_info( )	
if(video_signal_info_idx_present_flag && vps_num_video_signal_info_minus1 > 0)	
for(i = 1; i <= MaxLayersMinus1; i++)	
vps_video_signal_info_idx[i]	u(4)
vps_vui_bsp_hrd_present_flag	u(1)
if(vps_vui_bsp_hrd_present_flag)	
vps_vui_bsp_hrd_parameters( )	
}	

F.7.3.2.1.5 Video signal info syntax

video_signal_info( ) {	**Descriptor**
video_vps_format	u(3)
video_full_range_vps_flag	u(1)
colour_primaries_vps	u(8)
transfer_characteristics_vps	u(8)
matrix_coeffs_vps	u(8)
}	

F.7.3.2.1.6 VPS VUI bitstream partition HRD parameters syntax

vps_vui_bsp_hrd_parameters( ){	**Descriptor**
vps_num_bsp_hrd_parameters_minus1	ue(v)
for(i = 0; i <= vps_num_bsp_hrd_parameters_minus1; i++) {	
if(i > 0)	
bsp_cprms_present_flag[i]	u(1)
hrd_parameters(bsp_cprms_present_flag[i], vps_max_sub_layers_minus1)	
}	
for(h=1; h <= vps_num_layer_sets_minus1; h++) {	
num_bitstream_partitions[h]	ue(v)
for(i = 0; i < num_bitstream_partitions[h]; i++)	
for(j = 0; j <= vps_max_layers_minus1; j++)	
if(layer_id_included_flag[h][j])	
layer_in_bsp_flag[h][i][j]	u(1)
if(num_bitstream_partitions[h]) {	
num_bsp_sched_combinations[h]	ue(v)
for(i = 0; i < num_bsp_sched_combinations[h]; i++)	
for(j = 0; j < num_bitstream_partitions[h]; j++) {	
bsp_comb_hrd_idx[h][i][j]	ue(v)
bsp_comb_sched_idx[h][i][j]	ue(v)
}	
}	
}	
}	

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)
if(nuh_layer_id > 0) {	
update_rep_format_flag	u(1)
if(update_rep_format_flag)	
sps_rep_format_idx	u(8)
} else {	
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)
}	
if(nuh_layer_id = = 0) {	
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) {	
if(nuh_layer_id > 0)	

sps_infer_scaling_list_flag	u(1)
if(sps_infer_scaling_list_flag)	
sps_scaling_list_ref_layer_id	u(6)
else {	
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 [Ed. (GT): Syntax and semantics should be moved to base spec later.]	u(1)
if(sps_extension_flag) {	
for (i = 0; i < 8; i++)	
sps_extension_type_flag[i]	u(1)
if(sps_extension_type_flag[1])	
sps_multilayer_extension( )	
if(sps_extension_type_flag[7])	
while(more_rbsp_data( ))	
sps_extension_data_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

F.7.3.2.2.1 Sequence parameter set multilayer extension syntax

sps_multilayer_extension( ) {	**Descriptor**
inter_view_mv_vert_constraint_flag	u(1)
num_scaled_ref_layer_offsets	ue(v)
for(i = 0; i < num_scaled_ref_layer_offsets; i++) {	
scaled_ref_layer_id[i]	u(6)
scaled_ref_layer_left_offset[scaled_ref_layer_id[i]]	se(v)
scaled_ref_layer_top_offset[scaled_ref_layer_id[i]]	se(v)
scaled_ref_layer_right_offset[scaled_ref_layer_id[i]]	se(v)
scaled_ref_layer_bottom_offset[scaled_ref_layer_id[i]]	se(v)
}	
}	

F.7.3.2.3 Picture parameter set RBSP syntax

pic_parameter_set_rbsp( ) {	**Descriptor**
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
dependent_slice_segments_enabled_flag	u(1)
output_flag_present_flag	u(1)
num_extra_slice_header_bits	u(3)
sign_data_hiding_enabled_flag	u(1)
cabac_init_present_flag	u(1)
num_ref_idx_l0_default_active_minus1	ue(v)
num_ref_idx_l1_default_active_minus1	ue(v)
init_qp_minus26	se(v)
constrained_intra_pred_flag	u(1)
transform_skip_enabled_flag	u(1)
cu_qp_delta_enabled_flag	u(1)
if(cu_qp_delta_enabled_flag)	
diff_cu_qp_delta_depth	ue(v)
pps_cb_qp_offset	se(v)
pps_cr_qp_offset	se(v)
pps_slice_chroma_qp_offsets_present_flag	u(1)
weighted_pred_flag	u(1)
weighted_bipred_flag	u(1)
transquant_bypass_enabled_flag	u(1)
tiles_enabled_flag	u(1)
entropy_coding_sync_enabled_flag	u(1)
if(tiles_enabled_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
if(!uniform_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
row_height_minus1[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
pps_loop_filter_across_slices_enabled_flag	u(1)
deblocking_filter_control_present_flag	u(1)
if(deblocking_filter_control_present_flag) {	
deblocking_filter_override_enabled_flag	u(1)
pps_deblocking_filter_disabled_flag	u(1)
if(!pps_deblocking_filter_disabled_flag) {	
pps_beta_offset_div2	se(v)
pps_tc_offset_div2	se(v)
}	
}	

if(nuh_layer_id > 0)	
pps_infer_scaling_list_flag	u(1)
if(pps_infer_scaling_list_flag)	
pps_scaling_list_ref_layer_id	u(6)
else {	
pps_scaling_list_data_present_flag	u(1)
if(pps_scaling_list_data_present_flag)	
scaling_list_data( )	
}	
lists_modification_present_flag	u(1)
log2_parallel_merge_level_minus2	ue(v)
slice_segment_header_extension_present_flag	u(1)
pps_extension_flag	u(1)
if(pps_extension_flag)	
while(more_rbsp_data( ))	
pps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

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) {	
i = 0	
if(num_extra_slice_header_bits > i) {	
i++	
discardable_flag	u(1)
}	
if(num_extra_slice_header_bits > i) {	
i++	
cross_layer_bla_flag	u(1)
}	
if(num_extra_slice_header_bits > i) {	
i++	
poc_reset_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((nuh_layer_id > 0 && !poc_lsb_not_present_flag[LayerIdxInVPS[nuh_layer_id]]) || (nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP))	
slice_pic_order_cnt_lsb	u(v)
if(nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) {	
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 && !all_ref_layers_active_flag && 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)
if(NumActiveRefLayerPics != NumDirectRefLayers[nuh_layer_id])	
for(i = 0; i < NumActiveRefLayerPics; i++)	
inter_layer_pred_layer_idc[i]	u(v)
}	
}	
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 && NumPicTotalCurr > 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(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.

nal_unit_type specifies the type of RBSP data structure contained in the NAL unit as specified in Table 7 1.

When one picture picA of a layer layerA has nal_unit_type equal to TSA_N or TSA_R, each picture in the same access unit as picA in a direct or indirect reference layer of layerA shall have nal_unit_type equal to TSA_N or TSA_R.

When one picture picA of a layer layerA has nal_unit_type equal to STSA_N or STSA_R, each picture in the same access unit as picA in a direct or indirect reference layer of layerA shall have nal_unit_type equal to STSA_N or STSA_R.

nuh_layer_id specifies the identifier of the layer. The value of nuh_layer_id shall be in the range of 0 to 62, inclusive. The value of 63 may be specified in the future by ITU-T | ISO/IEC. Decoders shall ignore all data that follow the value 63 for nuh_layer_id in a NAL unit.

NOTE 3 – It is anticipated that in a future super multiview coding extension of this specification, the value of 63 for nuh_layer_id will be used to indicate an extended layer identifier.

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.

When nal_unit_type is equal to VPS_NUT, the value of nuh_layer_id shall be equal to 0. Decoder shall ignore NAL units with nal_unit_type equal to VPS_NUT and nuh_layer_id greater than 0.

When nal_unit_type is equal to PPS_NUT and the NAL unit contains the active PPS for a layer layerA with nuh_layer_id equal to nuhLayerIdA, the value of nuh_layer_id shall be equal to 0, nuhLayerIdA, or the nuh_layer_id of a direct or indirect reference layer of layerA.

When nal_unit_type is equal to SPS_NUT and the NAL unit contains the active SPS for a layer layerA with nuh_layer_id equal to nuhLayerIdA, the value of nuh_layer_id shall be equal to 0, nuhLayerIdA, or the nuh_layer_id of a direct or indirect reference layer of layerA.

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.

The contents of the hrd_parameters( ) syntax structure shall remain unchanged within a sequence of activated SPS RBSPs, in their activation order, from any activated SPS RBSP until the end of the bitstream or up to but excluding an SPS RBSP that is activated within the next access unit in which at least one of the following conditions is true:

– The access unit includes a picture for each nuh_layer_id value in TargetDecLayerIdList and each picture in the access unit is an IDR picture.

– The access unit includes an IRAP picture with nuh_layer_id equal to 0 for which NoClrasOutputFlag is equal to 1.

An activated VPS RBSP shall remain active until the end of the bitstream or until it is deactivated by another VPS RBSP in an access unit in which at least one of the following conditions is true:

– The access unit includes a picture for each nuh_layer_id value in TargetDecLayerIdList and each picture in the access unit is an IDR picture.

– The access unit includes an IRAP picture with nuh_layer_id equal to 0 for which NoClrasOutputFlag is equal to 1.

An activated SPS RBSP for a particular layer with nuh_layer_id greater than 0 shall remain active for a sequence of pictures in decoding order with that nuh_layer_id value starting from a picture, inclusive, that is an IRAP picture with NoRaslOutputFlag equal to 1 or for which FirstPicInLayerDecodedFlag[nuh_layer_id] is equal to 0, until the next picture, exclusive, that is an IRAP picture with NoRaslOutputFlag equal to 1 or for which FirstPicInLayerDecodedFlag[nuh_layer_id] is equal to 0.

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

During operation of the decoding process for NAL units of a non-base layer, the values of parameters of the active VPS RBSP, the active SPS RBSP for the non-base layer, and the active PPS RBSP for the non-base layer are considered in effect. For interpretation of SEI messages applicable to a coded picture of a non-base layer, the values of the active VPS RBSP, the active SPS RBSP for the non-base layer, and the active PPS RBSP for the non-base layer for the operation of the decoding process for the VCL NAL units of the coded picture are considered in effect unless otherwise specified in the SEI message semantics.

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.

– *Replace* "Each operation point is identified by the associated layer identifier list, denoted as OpLayerIdList, which consists of the list of nuh_layer_id values of all NAL units included in the operation point, in increasing order of nuh_layer_id values, and a variable OpTid, which is equal to the highest TemporalId of all NAL units included in the operation point." *with* "Each operation point is identified by the a list of nuh_layer_id values of all the pictures that are to be output, in increasing order of nuh_layer_id values, denoted as OptLayerIdList, and a variable OpTid, which is equal to the highest TemporalId of all NAL units included in the operation point. The layer identifier list associated with the list OptLayerIdList, denoted as OpLayerIdList, consists of the list of nuh_layer_id values of all NAL units included in the operation point, in increasing order of nuh_layer_id values.".

vps_max_layers_minus1 plus 1 specifies the maximum allowed number of layers in the CVS. vps_max_layers_minus1 shall be less than 63 in bitstreams conforming to this version of this Specification. The value of 63 for vps_max_layers_minus1 is reserved for future use by ITU-T | ISO/IEC. Although the value of vps_max_layers_minus1 is required to be less than 63 in this version of this Specification, decoders shall allow a value of vps_max_layers_minus1 equal to 63 to appear in the syntax.

NOTE 4 – It is anticipated that in a future super multiview coding extension of this specification, the value of 63 for vps_max_layers_minus1 will be used to indicate an extended number of layers.

The variable MaxLayersMinus1 is set equal to Min(62, vps_max_layers_minus1).

vps_max_layer_id specifies the maximum allowed value of nuh_layer_id of all NAL units in the CVS. vps_max_layer_id shall be less than 63 in bitstreams conforming to this version of this Specification. The value of 63 for vps_max_layer_id is reserved for future use by ITU-T | ISO/IEC. Although the value of vps_max_layer_id is required to be less than 63 in this version of this Specification, decoders shall allow a value of vps_max_layer_id equal to 63 to appear in the syntax.

vps_extension_offset specifies the byte offset, starting from the beginning of the VPS NAL unit, of the next set of fixed-length coded information starting from avc_base_layer_flag, when present, in the VPS NAL unit. When present, emulation prevention bytes that appear in the VPS NAL unit are counted for purposes of byte offset identification.

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 MaxLayersMinus1 is greater than 0, vps_extension_flag shall be equal to 1.

vps_extension_alignment_bit_equal_to_one 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

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

vps_vui_present_flag equal to 1 specifies that the vps_vui( ) syntax structure is present in the VPS. vps_vui_present_flag equal to 0 specifies that the vps_vui( ) syntax structure is not present in the VPS.

vps_vui_offset specifies the byte offset, starting from the beginning of the VPS NAL unit, of the set of fixed-length coded information starting from bit_rate_present_vps_flag, when present, in the VPS NAL unit. When present, emulation prevention bytes that appear in the VPS NAL unit are counted for purposes of byte offset identification.

splitting_flag equal to 1 indicates that the dimension_id[i][j] syntax elements are not present and that the binary representation of the nuh_layer_id value in the NAL unit header are split into NumScalabilityTypes segments with lengths, in bits, according to the values of dimension_id_len_minus1[j] and that the values of dimension_id[LayerIdxInVps[nuh_layer_id]][j] are inferred from the NumScalabilityTypes segments. splitting_flag equal to 0 indicates that the syntax elements dimension_id[i][j] are present.

NOTE 1 – When splitting_flag is equal to 1, scalable identifiers can be derived from the nuh_layer_id syntax element in the NAL unit header by a bit masked copy. 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_flag[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_flag[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	Reserved	
1	Multiview	View Order Index
2	Reserved	
3	Auxiliary	AuxId
4-15	Reserved	

NOTE 2 – It is anticipated that in future 3D extensions of this Specification, scalability mask index 0 will be used to indicate depth maps. It is anticipated that in future scalability extensions of this Specification, scalability mask index 2 will be used to indicate spatial/SNR scalability.

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

When splitting_flag is equal to 1, the following applies:

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

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

– 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.

It is a requirement of bitstream conformance that when NumScalabilityTypes is greater than 0, dimBitOffset[NumScalabilityTypes − 1] shall be less than 6.

vps_nuh_layer_id_present_flag equal to 1 specifies that layer_id_in_nuh[i] for i from 0 to MaxLayersMinus1, inclusive, are present. vps_nuh_layer_id_present_flag equal to 0 specifies that layer_id_in_nuh[i] for i from 0 to MaxLayersMinus1, inclusive, are not 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 the range of 0 to MaxLayersMinus1, inclusive, when layer_id_in_nuh[i] is not present, the value 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].

For i from 0 to MaxLayersMinus1, 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.

Depending on splitting_flag, the following applies:

– If splitting_flag is equal to 1, for i from 0 to MaxLayersMinus1, inclusive, and j from 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]).

– Otherwise (splitting_flag is equal to 0), for j from 0 to NumScalabilityTypes − 1, inclusive, dimension_id[0][j] is inferred to be equal to 0.

The variable ScalabilityId[i][smIdx] specifying the identifier of the smIdx-th scalability dimension type of the i-th layer, the variable ViewOrderIdx[layer_id_in_nuh[i]] specifying the view order index 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:

```
NumViews = 1
for( i = 0; i  <=  MaxLayersMinus1; i++ ) {
    lId = layer_id_in_nuh[ i ]
    for( smIdx= 0, j = 0; smIdx < 16; smIdx++ )
        if( scalability_mask_flag[ smIdx ] )
            ScalabilityId[ i ][ smIdx ] = dimension_id[ i ][ j++ ]
    ViewOrderIdx[ lId ] = ScalabilityId[ i ][ 1 ]
    if( i > 0  &&  ( ViewOrderIdx[ lId ]  !=  ScalabilityId[ i – 1][ 1 ] ) )
        NumViews++
    ViewScalExtLayerFlag[ lId ] = ( ViewOrderIdx[ lId ] > 0 )
    AuxId[ lId ] = ScalabilityId[ i ][ 3 ]
}
```

AuxId[lId] equal to 0 specifies the layer with nuh_layer_id equal to lId does not contain auxiliary pictures. AuxId[lId] greater than 0 specifies the type of auxiliary pictures in layer with nuh_layer_id equal to lId as specified in Table F-2.

Table F-2 – Mapping of AuxId to the type of auxiliary pictures

AuxId	Name of AuxId	Type of auxiliary pictures
1	AUX_ALPHA	Alpha plane
2	AUX_DEPTH	Depth picture
4-127		Reserved
128-143		Unspecified
144-255		Reserved

NOTE 3 – The interpretation of auxiliary pictures associated with AuxId in the range of 128 to 143, inclusive, is specified through means other than the AuxId value.

AuxId[lId] shall be in the range of 0 to 2, inclusive, or 128 to 143, inclusive, for bitstreams conforming to this version of this Specification. Although the value of AuxId[lId] shall be in the range of 0 to 2, inclusive, or 128 to 143, inclusive, in this version of this Specification, decoders shall allow values of AuxId[lId] in the range of 0 to 255, inclusive.

For an auxiliary picture with nuh_layer_id equal to nuhLayerIdA, an associated primary picture, if any, is the picture in the same access unit having AuxId[nuhLayerIdB] equal to 0 such that ScalabilityId[LayerIdxInVps[nuhLayerIdA]][j] is equal to ScalabilityId[LayerIdxInVps[nuhLayerIdB]][j] for all values of j in the range of 0 to 2, inclusive, and 4 to 15, inclusive.

It is a requirement of bitstream conformance that there shall be an associated primary picture for each auxiliary picture with AuxId[nuh_layer_id] equal to AUX_ALPHA.

NOTE 4 – It is not required that each auxiliary picture of each auxiliary picture type has an associated primary picture. For example, a layer with AuxId[nuh_layer_id] equal to AUX_DEPTH may represent a viewpoint of a range sensing camera, while the layers containing primary pictures may represent conventional cameras.

view_id_len specifies the length, in bits, of the view_id_val[i] syntax element. The value of view_id_len shall be greater than or equal to Ceil(Log2 (NumViews)). [Ed. (GT): Regarding that currently two different views are not required to have different view_id_val values the last constraint is not necessary.]

view_id_val[i] specifies the view identifier of the i-th view specified by the VPS. The length of the view_id_val[i] syntax element is view_id_len bits. When not present, the value of view_id_val[i] is inferred to be equal to 0.

For each layer with nuh_layer_id equal to nuhLayerId, the value ViewId[nuhLayerId] is set equal to view_id_val[ViewOrderIdx[nuhLayerId]].

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 MaxLayersMinus1, it is inferred to be equal to 0.

The variables NumDirectRefLayers[i] and RefLayerId[i][j] are derived as follows:

```
for( i = 0; i  <=  MaxLayersMinus1; i++ ) {
    iNuhLId = layer_id_in_nuh[ i ]
    NumDirectRefLayers[ iNuhLId ] = 0
    for( j = 0; j < i; j++ )
        if( direct_dependency_flag[ i ][ j ] )
            RefLayerId[ iNuhLId ][ NumDirectRefLayers[ iNuhLId ]++ ] = layer_id_in_nuh[ j ]
}
```

It is a requirement of bitstream conformance that AuxId[RefLayerId[nuhLayerIdA][j]] for any values of nuhLayerIdA and j shall be equal to AuxId[nuhLayerIdA], when AuxId[nuhLayerIdA] is in the range of 0 to 2, inclusive.

NOTE 5 – In other words, no prediction takes place between layers with a different value of AuxId, when AuxId is in the range of 0 to 2, inclusive.

vps_sub_layers_max_minus1_present_flag equal to 1 specifies that the syntax elements sub_layers_vps_max_minus1[i] are present. vps_sub_layers_max_minus1_present_flag equal to 0 specifies that the syntax elements sub_layers_vps_max_minus1[i] are not present.

sub_layers_vps_max_minus1[i] plus 1 specifies the maximum number of temporal sub-layers that may be present in the CVS for the layer with nuh_layer_id equal to layer_id_in_nuh[i]. The value of sub_layers_vps_max_minus1[i] shall be in the range of 0 to vps_max_sub_layers_minus1, inclusive. When not present, sub_layers_vps_max_minus1[i] is inferred to be equal to vps_max_sub_layers_minus1.

max_tid_ref_present_flag equal to 1 specifies that the syntax element max_tid_il_ref_pics_plus1[i][j] is present. max_tid_ref_present_flag equal to 0 specifies that the syntax element max_tid_il_ref_pics_plus1[i][j] is not present.

max_tid_il_ref_pics_plus1[i][j] 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 for pictures with nuh_layer_id equal to layer_id_in_nuh[j]. max_tid_il_ref_pics_plus1[i][j] 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][j] – 1 are not used as reference for inter-layer prediction for pictures with nuh_layer_id equal to layer_id_in_nuh[j]. When not present, max_tid_il_ref_pics_plus1[i][j] is inferred to be equal to 7.

all_ref_layers_active_flag equal to 1 specifies that for each picture referring to the VPS, the reference layer pictures that belong to all direct reference layers of the layer containing the picture and that might be used for inter-layer prediction as specified by the values of sub_layers_vps_max_minus1[i] and max_tid_il_ref_pics_plus1[i][j] are present in the same access unit as the picture and are included in the inter-layer reference picture set of the picture. all_ref_layers_active_flag equal to 0 specifies that the above restriction may or may not apply. [Ed. (GT): Consider renaming the syntax element, since not all reference layers are active anymore.]

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 or equal to 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_idc equal to 1 specifies that only the layer with the highest value of nuh_layer_id such that nuh_layer_id equal to nuhLayerIdA and AuxId[nuhLayerIdA] equal to 0 in each of the default output layer sets is a target output layer. default_one_target_output_layer_idc equal to 0 specifies that all layers in each of the default output layer sets are target output layers. default_one_target_output_layer_idc shall be equal to 0 or 1 in bitstreams conforming to this version of this Specification. Other values for default_one_target_output_layer_idc are reserved for future use by ITU-T | ISO/IEC. [Ed. GT: Should there be a default behaviour when reserved values appear?]

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.

The variable NumSubDpbs[i], specifying the number of sub-DPBs for the i-th output layer set, is set equal to NumLayersInIdList[i].

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.

alt_output_layer_flag affects picture output as specified in subclause F.13. [Ed. (GT) semantics should be more specific.]

NOTE 1 – When alt_output_layer_flag is equal to 0, pictures that are not at the target output layers are not output. When alt_output_layer_flag equal to 1 and a picture at the a target output layer is not present in an access unit, a picture with highest nuh_layer_id among those pictures of the access unit for which PicOutputFlag is equal to 1 and which are not among the target output layers is output.

rep_format_idx_present_flag equal to 1 specifies that the syntax elements vps_num_rep_formats_minus1 and vps_rep_format_idx[i] are present. rep_format_idx_present_flag equal to 0 specifies that the syntax elements vps_num_rep_formats_minus1 and vps_rep_format_idx[i] are not present.

vps_num_rep_formats_minus1 plus 1 specifies the number of the following rep_format( ) syntax structures in the VPS. When not present, the value of vps_num_rep_formats_minus1 is inferred to be equal to MaxLayersMinus1.

vps_rep_format_idx[i] specifies the index, into the list of rep_format( ) syntax structures in the VPS, of the rep_format( ) syntax structure that applies to the layer with nuh_layer_id equal to layer_id_in_nuh[i]. When not present, the value of vps_rep_format_idx[i] is inferred to be equal to (rep_format_idx_present_flag ? 0 : i). The value of vps_rep_format_idx[i] shall be in the range of 0 to vps_num_rep_formats_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.

poc_lsb_not_present_flag[i] equal to 1 specifies that the slice_pic_order_cnt_lsb syntax element is not present in the slice headers of IDR pictures with nuh_layer_id equal to layer_id_in_nuh[i] in the CVS. poc_lsb_not_present_flag[i] equal to 0 specifies that slice_pic_order_cnt_lsb syntax element may or may not be present in the slice headers of IDR pictures with nuh_layer_id equal to layer_id_in_nuh[i] in the CVS. When not present, poc_lsb_not_present_flag[i] is inferred to be equal to 0.

direct_dep_type_len_minus2 plus 2 specifies the number of bits of the direct_dependency_type[i][j] and the default_direct_dependency_type syntax elements. 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.

default_direct_dependency_flag equal to 1 specifies that the syntax element direct_dependency_type[i][j] is not present and inferred from default_direct_dependency_type. default_direct_dependency_flag equal to 0 indicates that the syntax element direct_dependency_type[i][j] is present.

default_direct_dependency_type, when present, specifies the inferred value of direct_dependency_type[i][j]. The length of the default_direct_dependency_type syntax element is direct_dep_type_len_minus2 + 2 bits. Although the value of default_direct_dependency_type is required to be in the range of 0 to 2, inclusive, in this version of this Specification, decoders shall allow values of default_direct_dependency_type in the range of 3 to $2^{32} - 2$, inclusive, to appear in the syntax.

direct_dependency_type[i][j] indicates the type of dependency between the layer with nuh_layer_id equal layer_id_in_nuh[i] and the layer with nuh_layer_id equal to layer_id_in_nuh[j]. direct_dependency_type[i][j] equal to 0 indicates that the layer with nuh_layer_id equal to layer_id_in_nuh[j] is used for both inter-layer sample prediction and inter-layer motion prediction of the layer with nuh_layer_id equal layer_id_in_nuh[i]. direct_dependency_type[i][j] equal to 1 indicates that the layer with nuh_layer_id equal to layer_id_in_nuh[j] is used for inter-layer sample prediction but not for inter-layer motion prediction of the layer with nuh_layer_id equal layer_id_in_nuh[i]. direct_dependency_type[i][j] equal to 2 indicates that the layer with nuh_layer_id equal to layer_id_in_nuh[j] is used for inter-layer motion prediction but not for inter-layer sample prediction of the layer with nuh_layer_id equal layer_id_in_nuh[i]. The length of the direct_dependency_type[i][j] syntax element is direct_dep_type_len_minus2 + 2 bits. 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. When not present, the value of direct_dependency_type[i][j] is inferred to be equal to default_direct_dependency_type.

The variables VpsInterLayerSamplePredictionEnabled[i][j] and VpsInterLayerMotionPredictionEnabled[i][j] are derived as follows:

VpsInterLayerSamplePredictionEnabled[i][j] = direct_dependency_type[i][j] & 0x1 (F-2)

VpsInterLayerMotionPredictionEnabled[i][j] = direct_dependency_type[i][j] & 0x2 (F-3)

[Ed. (JB): May need to define semantic constraints associated with values of VpsInterLayerSamplePredictionEnabled[i][j] and VpsInterLayerMotionPredictionEnabled[i][j].][Ed. (GT): VpsInterLayerSamplePredictionEnabled[i][j] and VpsInterLayerMotionPredictionEnabled[i][j] are currently not used.]

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.]

vps_vui_alignment_bit_equal_to_one shall be equal to 1.

F.7.4.3.1.2 Representation format semantics

chroma_and_bit_depth_vps_present_flag equal to 1 specifies that the syntax elements, chroma_format_vps_idc, bit_depth_vps_luma_minus8, and bit_depth_vps_chroma_minus8 are present and that the syntax element separate_colour_plane_vps_flag might be present. chroma_and_bit_depth_vps_present_flag equal to 0 specifies that the syntax elements, chroma_format_vps_idc, separate_colour_plane_vps_flag, bit_depth_vps_luma_minus8, and bit_depth_vps_chroma_minus8 are not present and inferred from the previous rep_format( ) syntax structure in the VPS.

The value of chroma_and_bit_depth_vps_present_flag of the first rep_format( ) syntax structure in the VPS shall be equal to 1.

pic_width_vps_in_luma_samples, **pic_height_vps_in_luma_samples**, **chroma_format_vps_idc**, **separate_colour_plane_vps_flag**, **bit_depth_vps_luma_minus8**, and **bit_depth_vps_chroma_minus8** are used for inference of the values of the SPS syntax elements pic_width_in_luma_samples, pic_height_in_luma_samples, chroma_format_idc, separate_colour_plane_flag, bit_depth_luma_minus8, and bit_depth_chroma_minus8, respectively, for each SPS that refers to the VPS. When not present in the i-th rep_format( ) syntax structure in the VPS, the value of each of these syntax elements is inferred to be equal to the value of the corresponding syntax element in the (i − 1)-th rep_format( ) syntax structure in the VPS. For each of these syntax elements, all constraints, if any, that apply to the value of the corresponding SPS syntax element also apply. [Ed. (GT) Consider explicit constraints here.].

F.7.4.3.1.3 DPB size semantics

sub_layer_flag_info_present_flag[i] equal to 1 specifies that sub_layer_dpb_info_present_flag[i][j] is present for i in the range of 1 to vps_max_sub_layers_minus1, inclusive. sub_layer_flag_info_present_flag[i] equal to 0 specifies that, for each value of j greater than 0, sub_layer_dpb_info_present_flag[i][j] is not present and the value is inferred to be equal to 0.

sub_layer_dpb_info_present_flag[i][j] equal to 1 specifies that max_vps_dec_pic_buffering_minus1[i][k][j] is present for k in the range of 0 to NumSubDpbs[i] – 1, inclusive, for the j-th sub-layer, and max_vps_num_reorder_pics[i][j] and max_vps_latency_increase_plus1[i][j] are present for the j-th sub-layer. sub_layer_dpb_info_present_flag[i][j] equal to 0 specifies that the values of max_vps_dec_pic_buffering_minus1[i][k][j] are equal to max_vps_dec_pic_buffering_minus1[i][k][j − 1] for k in the range of 0 to NumSubDpbs[i] – 1, inclusive, and that the values max_vps_num_reorder_pics[i][j] and max_vps_latency_increase_plus1[i][j] are set equal to max_vps_num_reorder_pics[i][j - 1] and max_vps_latency_increase_plus1[i][j - 1], respectively. The value of sub_layer_dpb_info_present_flag[i][0] for any possible value of i is inferred to be equal to 1.

max_vps_dec_pic_buffering_minus1[i][k][j] plus 1 specifies the maximum required size of the k-th sub-DPB for the CVS in the i-th output layer set in units of picture storage buffers when HighestTid is equal to j. When j is greater than 0, max_vps_dec_pic_buffering_minus1[i][k][j] shall be greater than or equal to max_vps_dec_pic_buffering_minus1[i][k][j - 1]. When max_vps_dec_pic_buffering_minus1[i][k][j] is not present for j in the range of 1 to vps_max_sub_layers_minus1 − 1, inclusive, it is inferred to be equal to max_vps_dec_pic_buffering_minus1[i][k][j - 1].

max_vps_num_reorder_pics[i][j] specifies, when HighestTid is equal to j, the maximum allowed number of access units containing a picture with PicOutputFlag equal to 1 that can precede any access unit auA that contains a picture with PicOutputFlag equal to 1 in the i-th output layer set in the CVS in decoding order and follow the access unit auA that contains a picture with PicOutputFlag equal to 1 in output order. When max_vps_num_reorder_pics[i][j] is not present for j in the range of 1 to vps_max_sub_layers_minus1 – 1, inclusive, due to sub_layer_dpb_info_present_flag[i][j] being equal to 0, it is inferred to be equal to max_vps_num_reorder_pics[i][j - 1].

max_vps_latency_increase_plus1[i][j] not equal to 0 is used to compute the value of VpsMaxLatencyPictures[i][j], which, when HighestTid is equal to j, specifies the maximum number of access units containing a picture with PicOutputFlag equal to 1 in the i-th output layer set that can precede any access unit auA that contains a picture with PicOutputFlag equal to 1 in the CVS in output order and follow the access unit auA that contains a picture with PicOutputFlag equal to 1 in decoding order. When max_vps_latency_increase_plus1[i][j] is not present for j in the range of 1 to vps_max_sub_layers_minus1 – 1, inclusive, due to sub_layer_dpb_info_present_flag[i][j] being equal to 0, it is inferred to be equal to max_vps_latency_increase_plus1[i][j - 1].

When max_vps_latency_increase_plus1[i][j] is not equal to 0, the value of VpsMaxLatencyPictures[i][j] is specified as follows:

$$\text{VpsMaxLatencyPictures[i][j]} = \text{max\_vps\_num\_reorder\_pics[i][j]} + \text{max\_vps\_latency\_increase\_plus1[i][j]} - 1 \quad \text{(F-4)}$$

When max_vps_latency_increase_plus1[i][j] is equal to 0, no corresponding limit is expressed. The value of max_vps_latency_increase_plus1[i][j] shall be in the range of 0 to $2^{32} - 2$, inclusive.

F.7.4.3.1.4 VPS VUI semantics

cross_layer_pic_type_aligned_flag equal to 1 specifies that within a CVS that refers to the VPS, all VCL NAL units that belong to an access unit have the same value of nal_unit_type. cross_layer_pic_type_aligned_flag equal to 0 specifies that within a CVS that refers to the VPS, all VCL NAL units in each access unit may or may not have the same value of nal_unit_type.

cross_layer_irap_aligned_flag equal to 1 specifies that IRAP pictures in the CVS are cross-layer aligned, i.e. when a picture pictureA of a layer layerA in an access unit is an IRAP picture, each picture pictureB in the same access unit that belongs to a direct reference layer of layerA or that belongs to a layer for which layerA is a direct reference layer of that layer is an IRAP picture and the VCL NAL units of pictureB have the same value of nal_unit_type as that of pictureA. cross_layer_irap_aligned_flag equal to 0 specifies that the above restriction may or may not apply. When not present, the value of cross_layer_irap_aligned_flag is inferred to be equal to 1. [Ed. (JB): Need to change the inference to be based on value of cross_layer_pic_type_aligned_flag, so that alignment is not inferred when the VPS VUI is not present.]

bit_rate_present_vps_flag equal to 1 specifies that the syntax element bit_rate_present_flag[i][j] is present. bit_rate_present_vps_flag equal to 0 specifies that the syntax element bit_rate_present_flag[i][j] is not present.

pic_rate_present_vps_flag equal to 1 specifies that the syntax element pic_rate_present_flag[i][j] is present. pic_rate_present_vps_flag equal to 0 specifies that the syntax element pic_rate_present_flag[i][j] is not present.

bit_rate_present_flag[i][j] equal to 1 specifies that the bit rate information for the j-th subset of the i-th layer set is present. bit_rate_present_flag[i] equal to 0 specifies that the bit rate information for the j-th subset of the i-th layer set is not present. The j-th subset of a layer set is the output of the sub-bitstream extraction process when it is invoked with the layer set, j, and the layer identifier list associated with the layer set as inputs. When not present, the value of bit_rate_present_flag[i][j] is inferred to be equal to 0.

pic_rate_present_flag[i][j] equal to 1 specifies that picture rate information for the j-th subset of the i-th layer set is present. pic_rate_present_flag[i][j] equal to 0 specifies that picture rate information for the j-th subset of the i-th layer set is not present. When not present, the value of pic_rate_present_flag[i][j] is inferred to be equal to 0.

avg_bit_rate[i][j] indicates the average bit rate of the j-th subset of the i-th layer set, in bits per second. The value is given by BitRateBPS(avg_bit_rate[i][j]) with the function BitRateBPS( ) being specified as follows:

$$\text{BitRateBPS}(x) = (x \,\&\, (2^{14} - 1)) * 10^{(2 + (x >> 14))} \quad \text{(F-5)}$$

The average bit rate is derived according to the access unit removal time specified in clause F.13. In the following, bTotal is the number of bits in all NAL units of the j-th subset of the i-th layer set, t_1 is the removal time (in seconds) of the first access unit to which the VPS applies, and t_2 is the removal time (in seconds) of the last access unit (in decoding order) to which the VPS applies. With x specifying the value of avg_bit_rate[i][j], the following applies:

– If t_1 is not equal to t_2, the following condition shall be true:

$$(x \,\&\, (2^{14} - 1)) == \text{Round}(\text{bTotal} \div ((t_2 - t_1) * 10^{(2 + (x >> 14))})) \quad \text{(F-6)}$$

– Otherwise (t_1 is equal to t_2), the following condition shall be true:

$$(x \,\&\, (2^{14} - 1)) == 0 \quad \text{(F-7)}$$

max_bit_rate_layer[i][j] indicates an upper bound for the bit rate of the j-th subset of the i-th layer set in any one-second time window of access unit removal time as specified in clause F.13. The upper bound for the bit rate in bits per second is given by BitRateBPS(max_bit_rate_layer[i][j]). The bit rate values are derived according to the access unit removal time specified in clause F.13. In the following, t_1 is any point in time (in seconds), t_2 is set equal to $t_1 + 1 \div 100$, and bTotal is the number of bits in all NAL units of access units with a removal time greater than or equal to t_1 and less than t_2. With x specifying the value of max_bit_rate_layer[i][j], the following condition shall be obeyed for all values of t_1:

$$(x \,\&\, (2^{14} - 1)) >= \text{bTotal} \div ((t_2 - t_1) * 10^{(2 + (x >> 14))}) \quad \text{(F-8)}$$

constant_pic_rate_idc[i][j] indicates whether the picture rate of the j-th subset of the i-th layer set is constant. In the following, a temporal segment tSeg is any set of two or more consecutive access units, in decoding order, of the j-th subset of the i-th layer set, auTotal(tSeg) is the number of access units in the temporal segment tSeg, t_1(tSeg) is the removal time (in seconds) of the first access unit (in decoding order) of the temporal segment tSeg, t_2(tSeg) is the removal time (in seconds) of the last access unit (in decoding order) of the temporal segment tSeg, and avgPicRate(tSeg) is the average picture rate in the temporal segment tSeg, and is specified as follows:

$$\text{avgPicRate}(\text{tSeg}) == \text{Round}(\text{auTotal}(\text{tSeg}) * 256 \div (t_2(\text{tSeg}) - t_1(\text{tSeg}))) \quad \text{(F-9)}$$

If the j-th subset of the i-th layer set only contains one or two access units or the value of avgPicRate(tSeg) is constant over all the temporal segments, the picture rate is constant; otherwise, the picture rate is not constant.

constant_pic_rate_idc[i][j] equal to 0 indicates that the picture rate of the j-th subset of the i-th layer set is not constant. constant_pic_rate_idc[i][j] equal to 1 indicates that the picture rate of the j-th subset of the i-th layer set is constant. constant_pic_rate_idc[i][j] equal to 2 indicates that the picture rate of the j-th subset of the i-th layer set may or may not be constant. The value of constant_pic_rate_idc[i][j] shall be in the range of 0 to 2, inclusive.

avg_pic_rate[i] indicates the average picture rate, in units of picture per 256 seconds, of the j-th subset of the layer set. With auTotal being the number of access units in the j-th subset of the i-th layer set, t_1 being the removal time (in seconds) of the first access unit to which the VPS applies, and t_2 being the removal time (in seconds) of the last access unit (in decoding order) to which the VPS applies, the following applies:

– If t_1 is not equal to t_2, the following condition shall be true:

$$\text{avg\_pic\_rate}[\ i\ ]\ = =\ \text{Round}(\ \text{auTotal} * 256 \div (\ t_2 - t_1\ )\ ) \quad \text{(F-10)}$$

– Otherwise (t_1 is equal to t_2), the following condition shall be true:

$$\text{avg\_pic\_rate}[\ i\ ]\ = =\ 0 \quad \text{(F-11)}$$

tiles_not_in_use_flag equal to 1 indicates that the value of tiles_enabled_flag is equal to 0 for each PPS that is referred to by at least one picture referring to the VPS. tiles_not_in_use_flag equal to 0 indicates that such a restriction may or may not apply.

tiles_in_use_flag[i] equal to 1 indicates that the value of tiles_enabled_flag is equal to 1 for each PPS that is referred to by at least one picture of the i-th layer specified by the VPS. tiles_in_use_flag[i] equal to 0 indicates that such a restriction may or may not apply.

loop_filter_not_across_tiles_flag[i] equal to 1 indicates that the value of loop_filter_across_tiles_enabled_flag is equal to 0 for each PPS that is referred to by at least one picture of the i-th layer specified by the VPS. loop_filter_not_across_tiles_flag[i] equal to 0 indicates that such a restriction may or may not apply.

tile_boundaries_aligned_flag[i][j] equal to 1 indicates that, when any two samples of one picture of the i-th layer specified by the VPS belong to one tile, the two collocated samples, when both present in the picture of the j-th direct reference layer of the i-th layer, belong to one tile, and when any two samples of one picture of the i-th layer belong to different tiles, the two collocated samples, when both present in the picture of the j-th direct reference layer of the i-th layer belong to different tiles. tile_boundaries_aligned_flag equal to 0 indicates that such a restriction may or may not apply. When not present, the value of tile_boundaries_aligned_flag[i][j] is inferred to be equal to 0.

wpp_not_in_use_flag equal to 1 indicates that the value of entropy_coding_sync_enabled_flag is equal to 0 for each PPS that is referred to by at least one picture referring to the VPS. wpp_not_in_use_flag equal to 0 indicates that such a restriction may or may not apply.

wpp_in_use_flag[i] equal to 1 indicates that the value of entropy_coding_sync_enabled_flag is equal to 1 for each PPS that is referred to by at least one picture of the i-th layer specified by the VPS. wpp_in_use_flag[i] equal to 0 indicates that such a restriction may or may not apply.

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

ilp_restricted_ref_layers_flag equal to 1 indicates that additional restrictions on inter-layer prediction as specified below apply for each direct reference layer of each layer specified by the VPS. ilp_restricted_ref_layers_flag equal to 0 indicates that additional restrictions on inter-layer prediction may or may not apply.

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

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

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

ctu_based_offset_enabled_flag[i][j] equal to 1 specifies that the spatial region, in units of CTUs, in each picture of the j-th direct reference layer of the i-th layer, that is not used for inter-layer prediction for decoding of any picture of the i-th layer is indicated by min_spatial_segment_offset_plus1[i][j] and min_horizontal_ctu_offset_plus1[i][j] together. ctu_based_offset_enabled_flag[i][j] equal to 0 specifies that the spatial region, in units of slice segments, tiles, or CTU rows, in each picture of the j-th direct reference layer of the i-th layer, that is not used for inter-layer prediction for decoding of any picture of the i-th layer 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][j], when ctu_based_offset_enabled_flag[i][j] is equal to 1, indicates the spatial region, in each picture of the j-th direct reference layer of the i-th layer, that is not used for inter-layer prediction for decoding of any picture of the i-th layer, together with min_spatial_segment_offset_plus1[i][j], as specified below. The value of min_horizontal_ctu_offset_plus1[i][j] shall be in the range of 0 to refPicWidthInCtbsY[i][j], inclusive.

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

minHorizontalCtbOffset[i][j] = (min_horizontal_ctu_offset_plus1[i][j] > 0) ? (F-12)
(min_horizontal_ctu_offset_plus1[i][j] – 1) : (refPicWidthInCtbsY[i][j] - 1)

The variables curCtbLog2SizeY[i], curPicWidthInCtbsY[i], and curPicHeightInCtbsY[i] are set equal to CtbLog2SizeY, PicWidthInCtbsY, and PicHeightInCtbsY, respectively, of the i-th layer.

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

xAddrOfCtb[i][j] = (ctbAddr % curPicWidthInCtbsY) << curCtbLog2SizeY (F-13)

yAddrOfCtb[i][j] = (ctbAddr / curPicWidthInCtbsY) << curCtbLog2SizeY (F-14)

xColCtb[i][j] = xAddrOfCtb[i][j] >> refCtbLog2SizeY[i][j] (F-15)

yColCtb[i][j] = yAddrOfCtb[i][j] >> refCtbLog2SizeY[i][j] (F-16)

colCtbAddr[i][j] = xColCtb[i][j] + (yColCtb[i][j] * refPicWidthInCtbsY[i][j]) (F-17)

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

− If ctu_based_offset_enabled_flag[i][j] is equal to 0, exactly one of the following applies:

  − In each PPS referred to by a picture in the j-th direct reference layer of the i-th 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 of a picture of the i-th layer 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 j-th direct reference layer of the i-th layer, and contains the CTU with raster scan address colCtbAddr[i][j]. 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 decoding 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 j-th direct reference layer of the i-th 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 of the i-th layer 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 j-th direct reference layer of the i-th layer and that contains the CTU with raster scan address colCtbAddr[i][j]. 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 decoding 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 j-th direct reference layer of the i-th 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 of the i-th layer 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 j-th direct reference layer of the i-th layer and that contains the CTU with raster scan address colCtbAddr[i][j]. 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][j] is equal to 1), the following applies:
  - The variable refCtbAddr[i][j] is derived as follows:

xOffset[i][j] =
((xColCtb[i][j] + minHorizontalCtbOffset[i][j]) > (refPicWidthInCtbsY[i][j] - 1)) ?
(refPicWidthInCtbsY[i][j] - 1 –xColCtb[i][j]) : (minHorizontalCtbOffset[i][j]) (F-18)

yOffset[i][j] = (min_spatial_segment_offset_plus1[i][j] – 1) * refPicWidthInCtbsY[i][j] (F-19)

refCtbAddr[i][j] = colCtbAddr[i][j] + xOffset[i][j] + yOffset[i][j] (F-20)

  - Let CTU A be any CTU in any picture picA of the i-th layer, 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 j-th direct reference layer of the i-th layer and that has raster scan address greater than refCtbAddr[i][j]. 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.

video_signal_info_idx_present_flag equal to 1 specifies that the syntax elements vps_num_video_signal_info_minus1, and vps_video_signal_info_idx[i] are present. video_signal_info_idx_present_flag equal to 0 specifies that the syntax elements vps_num_video_signal_info_minus1, and vps_video_signal_info_idx[i] are not present.

vps_num_video_signal_info_minus1 plus 1 specifies the number of the following video_signal_info( ) syntax structures in the VPS. When not present, the value of vps_num_video_signal_info_minus1 is inferred to be equal to MaxLayersMinus1.

vps_video_signal_info_idx[i] specifies the index, into the list of video_signal_info( ) syntax structures in the VPS, of the video_signal_info( ) syntax structure that applies to the layer with nuh_layer_id equal to layer_id_in_nuh[i]. When vps_video_signal_info_idx[i] is not present, vps_video_signal_info_idx[i] is inferred to be equal to (video_signal_info_idx_present_flag ? 0 : i). The value of vps_video_signal_info_idx[i] shall be in the range of 0 to vps_num_video_signal_info_minus1, inclusive.

vps_vui_bsp_hrd_present_flag equal to 0 specifies that no bitstream partition HRD parameters are present in the VPS VUI. vps_vui_bsp_hrd_present_flag equal to 1 specifies that bitstream partition HRD parameters are present in the VPS VUI.

F.7.4.3.1.5 Video signal info semantics

video_vps_format, **video_full_range_vps_flag**, **colour_primaries_vps**, **transfer_characteristics_vps**, **matrix_coeffs_vps** are used for inference of the values of the SPS VUI syntax elements video_format, video_full_range_flag, colour_primaries, transfer_characteristics, matrix_coeffs respectively, for each SPS that refers to the VPS.

For each of these syntax elements, all constraints, if any, that apply to the value of the corresponding SPS VUI syntax element also apply.

F.7.4.3.1.6 VPS VUI bitstream partition HRD parameters semantics

vps_num_bsp_hrd_parameters_minus1 plus 1 specifies the number of hrd_parameters( ) syntax structures present within the vps_vui_bsp_hrd_parameters( ) syntax structure.

bsp_cprms_present_flag[i] equal to 1 specifies that the HRD parameters that are common for all sub-layers are present in the i-th hrd_parameters( ) syntax structure in the vps_vui_bsp_hrd_parameters( ) syntax structure. bsp_cprms_present_flag[i] equal to 0 specifies that the HRD parameters that are common for all sub-layers are not present in the i-th hrd_parameters( ) syntax structure in the vps_vui_bsp_hrd_parameters( ) syntax structure and are derived to be the same as the (i − 1)-th hrd_parameters( ) syntax structure in the in the vps_vui_bsp_hrd_parameters( ) syntax structure. bsp_cprms_present_flag[0] is inferred to be equal to 1.

num_bitstream_partitions[h] specifies the number of bitstream partitions for which HRD parameters are specified for the layer set with index h.

layer_in_bsp_flag[h][i][j] specifies that the layer with index j is a part of bitstream partition with index i within the layer set with index h.

It is a requirement of bitstream conformance that bitstream partition with index j shall not include direct or indirect reference layers of any layers in bitstream partition i for any values of i and j in the range of 0 to num_bitstream_partitions[h] – 1, inclusive, such that i is less than j.

num_bsp_sched_combinations[h] specifies the number of combinations of delivery schedules and hrd_parameters( ) specified for bitstream partitions for the layer set with index h.

bsp_comb_hrd_idx[h][i][j] specifies the index of hrd_parameters( ) within the vps_vui_bsp_hrd_parameters( ) syntax structure used in the i-th combination of a delivery schedule and hrd_parameters( ) specified for the bitstream partition with index j and for the layer set with index h.

bsp_comb_sched_idx[h][i][j] specifies the index of a delivery schedule within the hrd_parameters( ) syntax structure with the index bsp_comb_hrd_idx[h][i][j] that is used in the i-th combination of a delivery schedule and hrd_parameters( ) specified for the bitstream partition with index j and for the layer set with index h.

F.7.4.3.2 Sequence parameter set RBSP semantics

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

NOTE 1 – All SPSs, regardless of the values of their nuh_layer_id, share the same value space for sps_seq_parameter_set_id. In other words, an SPS with nuh_layer_id equal to X and sps_seq_parameter_set_id equal to A would update the previously received SPS with nuh_layer_id not equal to X and sps_seq_parameter_set_id equal to A.

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.

update_rep_format_flag equal to 1 specifies that sps_rep_format_idx is present and that the sps_rep_format_idx-th rep_format( ) syntax structures in the active VPS applies to the layers that refer to this SPS. update_rep_format_flag equal to 0 specifies that sps_rep_format_idx is not present. When not present, the value of update_rep_format_flag is inferred to be equal to 0.

sps_rep_format_idx specifies the index, into the list of rep_format( ) syntax structures in the VPS, of the rep_format( ) syntax structure that applies to the layers that refer to this SPS. When not present, the value of sps_rep_format_idx is inferred to be equal to 0. The value of sps_rep_format_idx shall be in the range of 0 to vps_num_rep_formats_minus1, inclusive. [Ed. (GT): Inferences to 0 seems not to be necessary. We might consider to infer it to vps_rep_format_idx[LayerIdxInVps[layerIdCurr]], when not present.].

When a current picture with nuh_layer_id layerIdCurr greater than 0 refers to an SPS, the values of chroma_format_idc, separate_colour_plane_flag, pic_width_in_luma_samples, pic_height_in_luma_samples, bit_depth_luma_minus8, and bit_depth_chroma_minus8 are inferred or constrained as follows:

– The variable repFormatIdx is derived as follows:

  – If update_rep_format_flag is equal to 0, the variable repFormatIdx is set equal to vps_rep_format_idx[LayerIdxInVps[layerIdCurr]].

  – Otherwise, (update_rep_format_flag is equal to 1), the variable repFormatIdx is set equal to sps_rep_format_idx.

– If the nuh_layer_id of the active SPS for the layer with nuh_layer_id equal to layerIdCurr is equal to 0, the values of chroma_format_idc, separate_colour_plane_flag, pic_width_in_luma_samples, pic_height_in_luma_samples, bit_depth_luma_minus8, and bit_depth_chroma_minus8 are inferred to be equal to chroma_format_vps_idc, separate_colour_plane_vps_flag, pic_width_vps_in_luma_samples, pic_height_vps_in_luma_samples, bit_depth_vps_luma_minus8, and bit_depth_vps_chroma_minus8, respectively, of the repFormatIdx-th rep_format( ) syntax structure in the active VPS and the values of chroma_format_idc, separate_colour_plane_flag, pic_width_in_luma_samples, pic_height_in_luma_samples, bit_depth_luma_minus8, and bit_depth_chroma_minus8 of the active SPS for the layer with nuh_layer_id equal to layerIdCurr are ignored.

  NOTE 2 – The values are inferred from the VPS when a non-base layer refers to an SPS that is also referred to by the base layer, in which case the SPS has nuh_layer_id equal to 0. For the base layer, the values of these parameters in the active SPS for the base layer apply.

– Otherwise (the nuh_layer_id of the active SPS for the layer with nuh_layer_id equal to layerIdCurr is greater than zero), the following applies:

  – The values of chroma_format_idc, separate_colour_plane_flag, pic_width_in_luma_samples, pic_height_in_luma_samples, bit_depth_luma_minus8, and bit_depth_chroma_minus8 are inferred to be equal to chroma_format_vps_idc, separate_colour_plane_vps_flag, pic_width_vps_in_luma_samples, pic_height_vps_in_luma_samples, bit_depth_vps_luma_minus8, and bit_depth_vps_chroma_minus8, respectively, of the repFormatIdx-th rep_format( ) syntax structure in the active VPS.

– When update_rep_format_flag is equal to 1, it is a requirement of bitstream conformance that the value of chroma_format_idc, separate_colour_plane_flag, pic_width_in_luma_samples, pic_height_in_luma_samples, bit_depth_luma_minus8, or bit_depth_chroma_minus8 shall be less than or equal to chroma_format_vps_idc, separate_colour_plane_vps_flag, pic_width_vps_in_luma_samples, pic_height_vps_in_luma_samples, bit_depth_vps_luma_minus8, or bit_depth_vps_chroma_minus8, respectively, of the vps_rep_format_idx[j]-th rep_format( ) syntax structure in the active VPS, where j is equal to LayerIdxInVps[layerIdCurr].

chroma_format_idc specifies the chroma sampling relative to the luma sampling as specified in subclause 6.2. The value of chroma_format_idc shall be in the range of 0 to 3, inclusive. The value of chroma_format_idc shall be less than or equal to chroma_format_vps_idc. [Ed. (GT): These requirements seem to be redundant now. We should consider to remove them.]

It is a requirement of bitstream conformance that when AuxId[lId] is equal to AUX_ALPHA or AUX_DEPTH, chroma_format_idc shall be equal to 0 in the active SPS for the layer with nuh_layer_id equal to lId.

separate_colour_plane_flag equal to 1 specifies that the three colour components of the 4:4:4 chroma format are coded separately. separate_colour_plane_flag equal to 0 specifies that the colour components are not coded separately. When separate_colour_plane_flag is not present, it is inferred to be equal to 0. When separate_colour_plane_flag is equal to 1, the coded picture consists of three separate components, each of which consists of coded samples of one colour plane (Y, Cb, or Cr) and uses the monochrome coding syntax. In this case, each colour plane is associated with a specific colour_plane_id value. The value of separate_colour_plane_flag shall be less than or equal to separate_colour_plane_vps_flag

NOTE 3 – There is no dependency in decoding processes between the colour planes having different colour_plane_id values. For example, the decoding process of a monochrome picture with one value of colour_plane_id does not use any data from monochrome pictures having different values of colour_plane_id for inter prediction.

Depending on the value of separate_colour_plane_flag, the value of the variable ChromaArrayType is assigned as follows:

– If separate_colour_plane_flag is equal to 0, ChromaArrayType is set equal to chroma_format_idc.

– Otherwise (separate_colour_plane_flag is equal to 1), ChromaArrayType is set equal to 0.

pic_width_in_luma_samples specifies the width of each decoded picture in units of luma samples. pic_width_in_luma_samples shall not be equal to 0 and shall be an integer multiple of MinCbSizeY. The value of pic_width_in_luma_samples shall be less than or equal to pic_width_vps_in_luma_samples.

pic_height_in_luma_samples specifies the height of each decoded picture in units of luma samples. pic_height_in_luma_samples shall not be equal to 0 and shall be an integer multiple of MinCbSizeY. The value of pic_height_in_luma_samples shall be less than or equal to pic_height_vps_in_luma_samples.

bit_depth_luma_minus8 specifies the bit depth of the samples of the luma array $BitDepth_Y$ and the value of the luma quantization parameter range offset $QpBdOffset_Y$ as follows:

$$BitDepth_Y = 8 + bit\_depth\_luma\_minus8 \quad (F-21)$$

$$QpBdOffset_Y = 6 * bit\_depth\_luma\_minus8 \quad (F-22)$$

bit_depth_luma_minus8 shall be in the range of 0 to 6, inclusive. bit_depth_luma_minus8 shall be less than or equal to bit_depth_vps_luma_minus8.

bit_depth_chroma_minus8 specifies the bit depth of the samples of the chroma arrays $BitDepth_C$ and the value of the chroma quantization parameter range offset $QpBdOffset_C$ as follows:

$$BitDepth_C = 8 + bit\_depth\_chroma\_minus8 \quad (F-23)$$

$$QpBdOffset_C = 6 * bit\_depth\_chroma\_minus8 \quad (F-24)$$

bit_depth_chroma_minus8 shall be in the range of 0 to 6, inclusive. bit_depth_chroma_minus8 shall be less than or equal to bit_depth_vps_chroma_minus8.

sps_infer_scaling_list_flag equal to 1 specifies that the syntax elements of the scaling list data syntax structure of the SPS are inferred to be equal to those of the SPS that is active for the layer with nuh_layer_id equal to sps_scaling_list_ref_layer_id. sps_infer_scaling_list_flag equal to 0 specifies that the syntax elements of the scaling list data syntax structure are not inferred. When not present, the value of sps_infer_scaling_list_flag is inferred to be 0.

sps_scaling_list_ref_layer_id specifies the value of the nuh_layer_id of the layer for which the active SPS is associated with the same scaling list data as the current SPS.

The value of sps_scaling_list_ref_layer_id shall be in the range of 0 to 62, inclusive.

When avc_base_layer_flag is equal to 1, it is a requirement of bitstream conformance that the value of sps_scaling_list_ref_layer_id shall be greater than 0.

It is a requirement of bitstream conformance that, when an SPS with nuh_layer_id equal to nuhLayerIdA is active for a layer with nuh_layer_id equal to nuhLayerIdB and sps_infer_scaling_list_flag in the SPS is equal to 1, sps_infer_scaling_list_flag shall be equal to 0 for the SPS that is active for the layer with nuh_layer_id equal to sps_scaling_list_ref_layer_id. [Ed. (YK): This constraint is not necessarily needed. It would be nice to allow for all SPSs recursively infer the scaling list data from the lowest HEVC layer, when desirable, as that does not impose any additional decoder complexity anyway.]

It is a requirement of bitstream conformance that, when an SPS with nuh_layer_id equal to nuhLayerIdA is active for a layer with nuh_layer_id equal to nuhLayerIdB, the layer with nuh_layer_id equal to sps_scaling_list_ref_layer_id shall be a direct or indirect reference layer of the layer with nuh_layer_id equal to nuhLayerIdB.

sps_scaling_list_data_present_flag equal to 1 specifies that the scaling list data syntax structure is present in the SPS. sps_scaling_list_data_present_flag equal to 0 specifies that the scaling list data syntax structure is not present in the SPS. When not present, the value of sps_scaling_list_data_present_flag is inferred to be equal to 0.

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 4 – 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 1 specifies that sps_extension_type_flag[i] for i in the range of 0 to 7, inclusive are present in the SPS RBSP syntax structure. sps_extension_flag equal to 0 specifies that sps_extension_flag[i] for i in the range of 0 to 7, inclusive are not present in the SPS RBSP syntax structure.

sps_extension_type_flag[i] shall be equal to 0, for i equal to 0 and in the range of 2 to 6, inclusive, in bitstreams conforming to this version of this Specification. The value of 1 for sps_extension_type_flag[i], for i equal to 0 and in the range of 2 to 6, inclusive, is reserved for future use by ITU-T | ISO/IEC. sps_extension_type_flag[1] equal to 1 specifies that the sps_multilayer_extension syntax structure is present. sps_extension_type_flag[1] equal to 0 specifies that the sps_multilayer_extension syntax structure is not present. sps_extension_type_flag[7] equal to 0 specifies that no sps_extension_data_flag syntax elements are present in the SPS RBSP syntax structure. sps_extension_type_flag[7] shall be equal to 0 in bitstreams conforming to this version of this Specification. The value of 1 for sps_extension_type_flag[7] 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_extension_type_flag[7] in an SPS NAL unit.

[Ed. (GT) constraints on sps_extension_type_flag for i equal to 0 and in the range of 2 to 6 should be removed when semantics are moved to the base spec]

F.7.4.3.2.1 Sequence parameter set multilayer 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.

num_scaled_ref_layer_offsets specifies the number of sets of scaled reference layer offset parameters that are present in the SPS. The value of num_scaled_ref_layer_offsets shall be in the range of 0 to 62, inclusive. [Ed. (JB): Should consider if this constraint should be further restricted. Is there a limit on the number of direct reference layers? (MH): If that is desirable, we should specify the range like this: "in the range of 0 to highestActiveLayerId, inclusive, where the variable highestActiveLayerId is equal to the greatest value of nuh_layer_id of any picture for which this SPS is the active SPS".]

The i-th scaled reference layer offset parameters specify the spatial correspondence of a picture referring to this SPS relative to an associated inter-layer picture with nuh_layer_id equal to scaled_ref_layer_id[i]. If the layer with nuh_layer_id equal to scaled_ref_layer_id[i] is a direct reference layer of the current picture, the associated inter-layer picture is the picture that is or could be included in the reference picture lists of the current picture. Otherwise, the associated inter-layer picture is any picture with nuh_layer_id equal to scaled_ref_layer_id[i]. [Ed. (MH): If the term associated inter-layer picture becomes needed in other parts of the specification too, move the definition to F.3.]

NOTE 1 – When spatial scalability is in use, the associated inter-layer picture is a resampled picture of a direct reference layer.

NOTE 2 – scaled_ref_layer_id[i] need not be among the direct reference layers for example when the spatial correspondence of an auxiliary picture to its associated primary picture is specified.

scaled_ref_layer_id[i] specifies the nuh_layer_id value of the associated inter-layer picture for which scaled_ref_layer_left_offset[i], scaled_ref_layer_top_offset[i], scaled_ref_layer_right_offset[i] and scaled_ref_layer_bottom_offset[i] are specified. The value of scaled_ref_layer_id[i] shall be less than the nuh_layer_id of any layer for which this SPS is the active SPS. [Ed. (MH): A constraint that scaled reference offsets shall not be used for Stereo Main profile was added in the profile specification.]

scaled_ref_layer_left_offset[scaled_ref_layer_id[i]] specifies the horizontal offset between the top-left luma sample of the associated inter-layer picture with nuh_layer_id equal to scaled_ref_layer_id[i] and the top-left luma sample of the current picture in units of two luma samples. When not present, the value of scaled_ref_layer_left_offset[scaled_ref_layer_id[i]] is inferred to be equal to 0.

scaled_ref_layer_top_offset[scaled_ref_layer_id[i]] specifies the vertical offset between the top-left luma sample of the associated inter-layer picture with nuh_layer_id equal to scaled_ref_layer_id[i] and the top-left luma sample of the current picture in units of two luma samples. When not present, the value of scaled_ref_layer_top_offset[scaled_ref_layer_id[i]] is inferred to be equal to 0.

scaled_ref_layer_right_offset[scaled_ref_layer_id[i]] specifies the horizontal offset between the bottom-right luma sample of the associated inter-layer picture with nuh_layer_id equal to scaled_ref_layer_id[i] and the bottom-right luma sample of the current picture in units of two luma samples. When not present, the value of scaled_ref_layer_right_offset[scaled_ref_layer_id[i]] is inferred to be equal to 0.

scaled_ref_layer_bottom_offset[scaled_ref_layer_id[i]] specifies the vertical offset between the bottom-right luma sample of the associated inter-layer picture with nuh_layer_id equal to scaled_ref_layer_id[i] and the bottom-right luma sample of the current picture in units of two luma samples. When not present, the value of scaled_ref_layer_bottom_offset[scaled_ref_layer_id[i]] is inferred to be equal to 0.

F.7.4.3.3 Picture parameter set RBSP semantics

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

NOTE – All PPSs, regardless of the values of their nuh_layer_id, share the same value space for pps_pic_parameter_set_id. In other words, a PPS with nuh_layer_id equal to X and pps_pic_parameter_set_id equal to A would update the previously received PPS with nuh_layer_id not equal to X and pps_pic_parameter_set_id equal to A.

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 in the range of 0 to 2, inclusive, 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.

pps_infer_scaling_list_flag equal to 1 specifies that the syntax elements of the scaling list data syntax structure of the PPS are inferred to be equal to those of the PPS that is active for the layer with nuh_layer_id equal to pps_scaling_list_ref_layer_id. pps_infer_scaling_list_flag equal to 0 specifies that the syntax elements of the scaling list data syntax structure of the PPS are not inferred. When not present, the value of pps_infer_scaling_list_flag is inferred to be 0.

pps_scaling_list_ref_layer_id specifies the value of the nuh_layer_id of the layer for which the active PPS has the same scaling list data as the current PPS.

The value of pps_scaling_list_ref_layer_id shall be in the range of 0 to 62, inclusive.

When avc_base_layer_flag is equal to 1, it is a requirement of bitstream conformance that pps_scaling_list_ref_layer_id shall be greater than 0.

It is a requirement of bitstream conformance that, when a PPS with nuh_layer_id equal to nuhLayerIdA is active for a layer with nuh_layer_id equal to nuhLayerIdB and pps_infer_scaling_list_flag in the PPS is equal to 1, pps_infer_scaling_list_flag shall be equal to 0 for the PPS that is active for the layer with nuh_layer_id equal to pps_scaling_list_ref_layer_id.

It is a requirement of bitstream conformance that, when a PPS with nuh_layer_id equal to nuhLayerIdA is active for a layer with nuh_layer_id equal to nuhLayerIdB, the layer with nuh_layer_id equal to pps_scaling_list_ref_layer_id shall be a direct or indirect reference layer of the layer with nuh_layer_id equal to nuhLayerIdB.

pps_scaling_list_data_present_flag equal to 1 specifies that parameters are present in the PPS to modify the scaling lists specified by the active SPS. pps_scaling_list_data_present_flag equal to 0 specifies that the scaling list data used for the pictures referring to the PPS are inferred to be equal to those specified by the active SPS. When scaling_list_enabled_flag is equal to 0, the value of pps_scaling_list_data_present_flag shall be equal to 0. When scaling_list_enabled_flag is equal to 1, sps_scaling_list_data_present_flag is equal to 0, and pps_scaling_list_data_present_flag is equal to 0, the default scaling list data are used to derive the array ScalingFactor as described in the scaling list data semantics 7.4.5.

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 and additions.

When present, the value of the slice segment header syntax elements slice_pic_parameter_set_id, pic_output_flag, no_output_of_prior_pics_flag, slice_pic_order_cnt_lsb, short_term_ref_pic_set_sps_flag, short_term_ref_pic_set_idx, num_long_term_sps, num_long_term_pics, slice_temporal_mvp_enabled_flag, discardable_flag, cross_layer_bla_flag, poc_reset_flag, inter_layer_pred_enabled_flag, and num_inter_layer_ref_pics_minus1 shall be the same in all slice segment headers of a coded picture. When present, the value of the slice segment header syntax elements lt_idx_sps[i], poc_lsb_lt[i], used_by_curr_pic_lt_flag[i], delta_poc_msb_present_flag[i], delta_poc_msb_cycle_lt[i], and inter_layer_pred_layer_idc[i] shall be the same in all slice segment headers of a coded picture for each possible value of i.

– "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.

cross_layer_bla_flag equal to 1 affects the derivation of NoClrasOutputFlag as specified in clause 8.1. cross_layer_bla_flag shall be equal to 0 for pictures with nal_unit_type not equal to IDR_W_RADL or IDR_N_LP or with nuh_layer_id not equal to 0.

poc_reset_flag equal to 1 specifies that the derived picture order count for the current picture is equal to 0. poc_reset_flag equal to 0 specifies that the derived picture order count for the current picture may or may not be equal to 0. When not present, the value of poc_reset_flag is inferred to be equal to 0.

NOTE – When poc_reset_flag is equal to 1 in a base-layer picture, PicOrderCntVal is derived differently depending on whether the decoding process of subclause 8.3.1 or subclause F.8.3.1 is applied. Furthermore, when a base-layer picture with poc_reset_flag equal to 1 is prevTid0Pic according to subclause 8.3.1 or F.8.3.1, the variable prevPicOrderCntLsb is derived differently in subclauses 8.3.1 and F.8.3.1. In order to avoid PicOrderCntMsb to be updated incorrectly in one of the subclauses 8.3.1 or F.8.3.1, when prevTid0Pic is a base-layer picture with poc_reset_flag equal to 1 and either of the following conditions is true for prevPicOrderCntLsb derived with one of the subclauses 8.3.1 or F.8.3.1, the value of pic_order_cnt_lsb of prevTid0Pic shall be such that the same condition is true also for prevPicOrderCntLsb derived with the other one of the subclauses 8.3.1 or F.8.3.1:

– (slice_pic_order_cnt_lsb < prevPicOrderCntLsb) &&
((prevPicOrderCntLsb − slice_pic_order_cnt_lsb) >= (MaxPicOrderCntLsb / 2))

– (slice_pic_order_cnt_lsb > prevPicOrderCntLsb) &&
((slice_pic_order_cnt_lsb – prevPicOrderCntLsb) > (MaxPicOrderCntLsb / 2))

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.

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 variables numRefLayerPics and refLayerPicFlag[i] and refLayerPicIdc[j] are derived as follows:

```
for( i = 0, j = 0;  i < NumDirectRefLayers[ nuh_layer_id ]; i++ ) {
    refLayerIdx = LayerIdxInVps[ RefLayerId[ nuh_layer_id ][ i ] ]
    refLayerPicFlag[ i ] = ( sub_layers_vps_max_minus1[ refLayerIdx ] >= TemporalId )  &&
                  ( max_tid_il_ref_pics_plus1[ refLayerIdx ][ LayerIdxInVps[ nuh_layer_id ] ] > TemporalId )
    if( refLayerPicFlag[ i ] )
        refLayerPicIdc[ j++ ] = i
}

numRefLayerPics = j
```

The variable NumActiveRefLayerPics is derived as follows:

```
if( nuh_layer_id  = =  0  | |  NumDirectRefLayers[ nuh_layer_id ]  = =  0 )
    NumActiveRefLayerPics = 0
else if( all_ref_layers_active_flag )
    NumActiveRefLayerPics = numRefLayerPics
else if( !inter_layer_pred_enabled_flag )
    NumActiveRefLayerPics = 0
else if( max_one_active_ref_layer_flag  | |  NumDirectRefLayers[ nuh_layer_id ]  = = 1 )
    NumActiveRefLayerPics = refLayerPicFlag[ 0 ] ? 1 : 0
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 refLayerPicIdc[i].

When i is greater than 0, inter_layer_pred_layer_idc[i] shall be greater than inter_layer_pred_layer_idc[i − 1].

The variables RefPicLayerId[i] for all values of i in the range of 0 to NumActiveRefLayerPics − 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 ] ]
```

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]]][LayerIdxInVps[nuh_layer_id]] is greater than TemporalId.

– The values of max_tid_il_ref_pics_plus1[LayerIdxInVps[RefPicLayerId[i]]][LayerIdxInVps[nuh_layer_id]] 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.

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 NumPicTotalCurr is replaced by:

```
NumPicTotalCurr = 0
for( i = 0; i < NumNegativePics[ CurrRpsIdx ]; i++)
  if(UsedByCurrPicS0[ CurrRpsIdx ][ i ] )
      NumPicTotalCurr++
for( i = 0; i < NumPositivePics[ CurrRpsIdx ]; i++)                    (F-25)
  if(UsedByCurrPicS1[ CurrRpsIdx ][ i ] )
      NumPicTotalCurr++
for( i = 0; i < num_long_term_sps + num_long_term_pics; i++ )
  if( UsedByCurrPicLt[ i ] )
      NumPicTotalCurr++
NumPicTotalCurr  +=  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 changes:

– Replace the references to clause 7, clause 10, and subclause 8.1.1 with subclauses F.7, F.10, and F.8.1.1, respectively.

– Add at the end of the subclause, add the following sentence:

When the current picture has nuh_layer_id greater than 0, the decoding process for a coded picture with nuh_layer_id greater than 0 as specified in subclause F.8.1.2 is invoked.

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

The specifications in subclause 8.1.1 apply with the following changes:

– Replace the references to subclauses 8.2, 8.3, 8.3.1, 8.3.2, 8.3.3, 8.3.4, 8.4, 8.5, 8.6, and 8.7 with subclauses F.8.2, F.8.3, F.8.3.1, F.8.3.2, F.8.3.3, F.8.3.4, F.8.4, F.8.5, F.8.6, and F.8.7, respectively.

F.8.1.2 Decoding process for a coded picture with nuh_layer_id greater than 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.8.1.3 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. (YK): It looks a bit odd to refer to Annex G here. Is this avoidable?]

– 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.8.1.4 is invoked.

F.8.1.3 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 F.8.2.

2. The processes in subclause F.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 F.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 F.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 FirstPicInLayerDecodedFlag[nuh_layer_id] is equal to 0, the decoding process for generating unavailable reference pictures specified in subclause F.8.1.5 is invoked, which needs to be invoked only for the first slice segment of a picture.

   – When FirstPicInLayerDecodedFlag[nuh_layer_id] is not equal to 0 and the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, the decoding process for generating unavailable reference pictures specified in subclause F.8.3.3 is invoked, which needs to be invoked only for the first slice segment of a picture.

F.8.1.4 Decoding process for ending the decoding of a coded picture with nuh_layer_id greater than 0

PicOutputFlag is set as follows:

– If LayerInitializedFlag[nuh_layer_id] is equal to 0, PicOutputFlag is set equal to 0.

– Otherwise, 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".

– 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.8.1.4.1 is invoked with latestDecLayerId equal to nuh_layer_id as input.

When FirstPicInLayerDecodedFlag[nuh_layer_id] is equal to 0, FirstPicInLayerDecodedFlag[nuh_layer_id] is set equal to 1.

F.8.1.4.1 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 – 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
iLidx = LayerIdxInVps[ TargetDecLayerIdList[ i ] ]
    for( j = latestDecIdx + 1; j < numTargetDecLayers; j++ ) {
        jLidx = LayerIdxInVps[ TargetDecLayerIdList[ j ] ]
        if( currTid  <=  ( max_tid_il_ref_pics_plus1[ iLidx ][ jLidx ] –1 ) )
            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.8.1.5 Generation of unavailable reference pictures for pictures first in decoding order within a layer

This process is invoked for a picture with nuh_layer_id equal to layerId, when FirstPicInLayerDecodedFlag[layerId] is equal to 0.

NOTE – A cross-layer random access skipped (CL-RAS) picture is a picture with nuh_layer_id equal to layerId such that LayerInitializedFlag[layerId] is equal to 0 when the decoding process for starting the decoding of a coded picture with nuh_layer_id greater than 0 is invoked. The entire specification of the decoding process for CL-RAS pictures is included only for purposes of specifying constraints on the allowed syntax content of such CL-RAS pictures. During the decoding process, any CL-RAS pictures may be ignored, as these pictures are not specified for output and have no effect on the decoding process of any other pictures that are specified for output. However, in HRD operations as specified in Annex C, CL-RAS pictures may need to be taken into consideration in derivation of CPB arrival and removal times.

When this process is invoked, the following applies:

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

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

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

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

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

   - The value of nuh_layer_id for the generated picture is set equal to nuh_layer_id.
- For each RefPicSetStCurrAfter[i], with i in the range of 0 to NumPocStCurrAfter – 1, inclusive, that is equal to “no-reference picture”, a picture is generated as specified in subclause 8.3.3.2, and the following applies:
   - The value of PicOrderCntVal for the generated picture is set equal to PocStCurrAfter[i].
   - The value of PicOutputFlag for the generated picture is set equal to 0.
   - The generated picture is marked as "used for short-term reference".
   - RefPicSetStCurrAfter[i] is set to be the generated reference picture.
   - The value of nuh_layer_id for the generated picture is set equal to nuh_layer_id.
- For each RefPicSetStFoll[i], with i in the range of 0 to NumPocStFoll − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in subclause 8.3.3.2, and the following applies:
   - The value of PicOrderCntVal for the generated picture is set equal to PocStFoll[i].
   - The value of PicOutputFlag for the generated picture is set equal to 0.
   - The generated picture is marked as "used for short-term reference".
   - RefPicSetStFoll[i] is set to be the generated reference picture.
   - The value of nuh_layer_id for the generated picture is set equal to nuh_layer_id.
- For each RefPicSetLtCurr[i], with i in the range of 0 to NumPocLtCurr – 1, inclusive, that is equal to “no-reference picture”, a picture is generated as specified in subclause 8.3.3.2, and the following applies:
   - The value of PicOrderCntVal for the generated picture is set equal to PocLtCurr[i].
   - The value of slice_pic_order_cnt_lsb for the generated picture is inferred to be equal to (PocLtCurr[i] & (MaxPicOrderCntLsb – 1)).
   - The value of PicOutputFlag for the generated picture is set equal to 0.
   - The generated picture is marked as "used for long-term reference".
   - RefPicSetLtCurr[i] is set to be the generated reference picture.
   - The value of nuh_layer_id for the generated picture is set equal to nuh_layer_id.
- For each RefPicSetLtFoll[i], with i in the range of 0 to NumPocLtFoll − 1, inclusive, that is equal to "no reference picture", a picture is generated as specified in subclause 8.3.3.2, and the following applies:
   - The value of PicOrderCntVal for the generated picture is set equal to PocLtFoll[i].
   - The value of slice_pic_order_cnt_lsb for the generated picture is inferred to be equal to (PocLtFoll[i] & (MaxPicOrderCntLsb – 1)).
   - The value of PicOutputFlag for the generated picture is set equal to 0.
   - The generated picture is marked as "used for long-term reference".
   - RefPicSetLtFoll[i] is set to be the generated reference picture.
   - The value of nuh_layer_id for the generated picture is set equal to nuh_layer_id.

F.8.2 NAL unit decoding process

The specifications in subclause 8.2 apply.

F.8.3 Slice decoding processes

F.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.

If FirstPicInLayerDecodedFlag[nuh_layer_id] is equal to 0 or the current picture is an IRAP picture with NoRaslOutputFlag equal to 1, the variable PicOrderCntMsb is set equal to 0. [Ed. (MH): When the first picture in an enhancement layer is in an access unit which follows in decoding order and precedes in output order an initial IRAP access unit with NoClrasOutputFlag equal to 1, PicOrderCntVal of the first picture in the enhancement layer differs from the PicOrderCntVal of the base-layer picture in the same access unit.] [Ed. (MH): This derivation of PicOrderCntMsb equal to 0 imposes a constraint that the layer-wise start-up up to the highest layer must take place within a POC range of 0 to MaxPicOrderLsb – 1, inclusive.] Otherwise, PicOrderCntMsb is derived as follows:

– The variable prevPicOrderCntLsb is set equal to PrevPicOrderCnt[nuh_layer_id] & (MaxPicOrderCntLsb − 1).

– The variable prevPicOrderCntMsb is set equal to PrevPicOrderCnt[nuh_layer_id] − prevPicOrderCntLsb.

– PicOrderCntMsb is derived as follows:

```
if( ( slice_pic_order_cnt_lsb < prevPicOrderCntLsb ) &&
        ( ( prevPicOrderCntLsb − slice_pic_order_cnt_lsb ) >= ( MaxPicOrderCntLsb / 2 ) ) )
    PicOrderCntMsb = prevPicOrderCntMsb + MaxPicOrderCntLsb                    (F-26)
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 (F-27)

When poc_reset_flag is equal to 1, the following steps apply in the order listed:

– The PicOrderCntVal of each picture that is in the DPB and belongs to the same layer as the current picture is decremented by PicOrderCntVal.

– PrevPicOrderCnt[nuh_layer_id] is decremented by PicOrderCntVal.

– PicOrderCntVal is set equal to 0.

When the current picture is not a RASL picture, a RADL picture or a sub-layer non-reference picture, and the current picture has TemporalId equal to 0, PrevPicOrderCnt[nuh_layer_id] is set equal to PicOrderCntVal.

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 in the same layer shall not be the same.

The function PicOrderCnt(picX) is specified as follows:

PicOrderCnt(picX) = PicOrderCntVal of the picture picX (F-28)

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

DiffPicOrderCnt(picA, picB) = PicOrderCnt(picA) − PicOrderCnt(picB) (F-29)

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 – 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.

F.8.3.2 Decoding process for reference picture set

The specifications in subclause 8.3.2 apply with the following changes:

– Replace the references to subclauses 7.4.7.2, 8.3.1, 8.3.3, and 8.3.4 to subclauses F.7.4.7.2, F.8.3.1, F.8.3.3, and F.8.3.4, respectively.

F.8.3.3 Decoding process for generating unavailable reference pictures

The specifications in subclause 8.3.3 apply.

F.8.3.4 Decoding process for reference picture lists construction

The specifications in subclause 8.3.4 apply.

F.8.3.5 Decoding process for collocated picture and no backward prediction flag

The specifications in subclause 8.3.5 apply.

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.

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)

F.12 Byte stream format

The specifications in Annex B apply.

F.13 Hypothetical reference decoder

The specifications in Annex C and its subclauses apply.

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 Layers not present SEI message syntax

layers_not_present(payloadSize) {	**Descriptor**
lp_sei_active_vps_id	u(4)
for(i = 0; i <= MaxLayersMinus1; i++)	
layer_not_present_flag[i]	u(1)
}	

F.14.1.2 Inter-layer constrained tile sets SEI message syntax

inter_layer_constrained_tile_sets(payloadSize) {	**Descriptor**
il_all_tiles_exact_sample_value_match_flag	u(1)
il_one_tile_per_tile_set_flag	u(1)
if(!il_one_tile_per_tile_set_flag) {	
il_num_sets_in_message_minus1	ue(v)
if(il_num_sets_in_message_minus1)	
skipped_tile_set_present_flag	u(1)
numSignificantSets = il_num_sets_in_message_minus1 – skipped_tile_set_present_flag + 1	
for(i = 0; i < numSignificantSets; i++) {	
ilcts_id[i]	ue(v)
il_num_tile_rects_in_set_minus1[i]	ue(v)
for(j = 0; j <= il_num_tile_rects_in_set_minus1[i]; j++) {	
il_top_left_tile_index[i **][** j **]**	ue(v)
il_bottom_right_tile_index[i]**[** j **]**	ue(v)
}	
ilc_idc[i]	u(2)
if (!il_all_tiles_exact_sample_value_match_flag)	
il_exact_sample_value_match_flag[i]	u(1)
}	
} else	
all_tiles_ilc_idc	u(2)
}	

F.14.1.3 Bitstream partition nesting SEI message syntax

bsp_nesting(payloadSize) {	**Descriptor**
bsp_idx	ue(v)
while(!byte_aligned( ))	
bsp_nesting_zero_bit /* equal to 0 */	u(1)
do	
sei_message( )	
while(more_rbsp_data( ))	
}	

F.14.1.4 Bitstream partition initial arrival time SEI message syntax

bsp_initial_arrival_time(payloadSize) {	**Descriptor**
if(NalHrdBpPresentFlag)	
for(i = 0; i < SchedCombCnt; i++)	
nal_initial_arrival_delay[i]	u(v)
else	
for(i = 0; i < SchedCombCnt; i++)	
vcl_initial_arrival_delay[i]	u(v)
}	

F.14.1.5 Bitstream partition HRD parameters SEI message syntax

bsp_hrd(payloadSize) {	Descriptor
sei_num_bsp_hrd_parameters_minus1	ue(v)
for(i = 0; i <= sei_num_bsp_hrd_parameters_minus1; i++) {	
if(i > 0)	
sei_bsp_cprms_present_flag[i]	u(1)
hrd_parameters(sei_bsp_cprms_present_flag[i], nesting_max_temporal_id_plus1[0] – 1)	
}	
for(h=0; h <= nesting_num_ops_minus1; h++) {	
lsIdx = nesting_op_idx[h]	
sei_num_bitstream_partitions_minus1[lsIdx]	ue(v)
for(i = 0; i <= sei_num_bitstream_partitions_minus1[lsIdx]; i++)	
for(j = 0; j <= vps_max_layers_minus1; j++)	
if(layer_id_included_flag[nesting_op_idx[lsIdx]][j])	
sei_layer_in_bsp_flag[lsIdx][i][j]	u(1)
sei_num_bsp_sched_combinations_minus1[lsIdx]	ue(v)
for(i = 0; i <= sei_num_bsp_sched_combinations_minus1[lsIdx]; i++)	
for(j = 0; j <= sei_num_bitstream_partitions_minus1[lsIdx]; j++) {	
sei_bsp_comb_hrd_idx[lsIdx][i][j]	ue(v)
sei_bsp_comb_sched_idx[lsIdx][i][j]	ue(v)
}	
}	
}	

F.14.2 SEI message semantics

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

SEI message	Persistence scope
Layers not 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 not present SEI message or the end of the CVS, whichever is earlier in decoding order
Inter-layer constrained tile sets	The CVS containing the SEI message
Bitstream partition nesting	Depending on the nested SEI messages. Each nested SEI message has the same persistence scope as if the SEI message was not nested
Bitstream partition initial arrival time	The remainder of the bitstream partition (specified by the containing bitstream partition nesting SEI message)
Bitstream partition HRD parameters	The CVS containing the SEI message

The constraints of bitstream conformance specified in clause D.3.1 apply with the following additions.

Let prevVclNalUnitInAu of an SEI NAL unit or an SEI message be the preceding VCL NAL unit in decoding order, if any, in the same access unit, and nextVclNalUnitInAu of an SEI NAL unit or an SEI message be the next VCL NAL unit in decoding order, if any, in the same access unit. It is a requirement of bitstream conformance that the following restrictions apply:

– When a bitstream partition HRD parameters SEI message contained in a scalable nesting SEI message is present in an access unit, the scalable nesting SEI message shall not follow any other SEI message that follows the prevVclNalUnitInAu of the scalable nesting SEI message and precedes the nextVclNalUnitInAu of the scalable nesting SEI message, other than an active parameter sets SEI message, a non-nested buffering period SEI message, a non-nested picture timing SEI message, a non-nested decoding unit information SEI message, a scalable nesting SEI message including a buffering period SEI message, a picture timing SEI message or a decoding unit information SEI message, or another scalable nesting SEI message that contains a bitstream partition HRD parameters SEI message.

– When a buffering period SEI message, a picture timing SEI message, a decoding unit information SEI message or a bitstream partition initial arrival time SEI message is present in a bitstream partition nesting SEI message contained in a scalable nesting SEI message, the scalable nesting SEI message shall not follow any other SEI message that follows the prevVclNalUnitInAu of the scalable nesting SEI message and precedes the nextVclNalUnitInAu of the scalable nesting SEI message, other than an active parameter sets SEI message, a non-nested buffering period SEI message, a non-nested picture timing SEI message, a non-nested decoding unit information SEI message, a scalable nesting SEI message including a buffering period SEI message, a picture timing SEI message or a decoding unit information SEI message, a scalable nesting SEI message including a bitstream partition HRD parameters SEI message, or another scalable nesting SEI message that contains a bitstream partition nesting SEI message including a buffering period SEI message, a picture timing SEI message, a decoding unit information SEI message or a bitstream partition initial arrival time SEI message.

F.14.2.1 Layers not present SEI message semantics

The layers not present SEI message provides a mechanism for signalling that VCL NAL units of particular layers indicated by the VPS are not present in a particular set of access units.

The target access units are defined as the set of access units starting from the access unit containing the layers not present SEI message up to but not including the next access unit, in decoding order, that contains a layers not present change SEI message or the end of the CVS, whichever is earlier in decoding order.

When present, the layers not present SEI message applies to the target access units.

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

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

lp_sei_active_vps_id identifies the active VPS of the CVS containing the layers not present SEI message. 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_not_present_flag[i] equal to 1 indicates that there are no VCL NAL units with nuh_layer_id equal to layer_id_in_nuh[i] present in the target access units. layer_not_present_flag[i] equal to 0 indicates that there may or may not be VCL NAL units with nuh_layer_id equal to layer_id_in_nuh[i] present in the target access units.

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

F.14.2.2 Inter-layer constrained tile sets SEI message semantics

The scope of the inter-layer constrained tile sets SEI message is the complete CVS. When an inter-layer tile sets SEI message is present in any access unit of a CVS, it shall be present for the first access unit of the CVS in decoding order and may also be present for other access units of the CVS.

The inter-layer constrained tile sets SEI message shall not be present for a layer when tiles_enabled_flag is equal to 0 for any PPS that is active for the layer.

The inter-layer constrained tile sets SEI message shall not be present for a layer unless every PPS that is active for the layer has tile_boundaries_aligned_flag equal to 1 or fulfills the conditions that would be indicated by tile_boundaries_aligned_flag being equal to 1.

The presence of the inter-layer tile sets SEI message indicates that the inter-layer inter prediction process is constrained such that no sample value outside each identified tile set, and no sample value at a fractional sample position that is derived using one or more sample values outside the identified tile set, is used for inter prediction of any sample within the identified tile set.

NOTE 1 – When loop filtering and resampling filter is applied across tile boundaries, inter-layer prediction of any samples within an inter-layer constrained tile set that refers to samples within 8 samples from an inter-layer constrained tile set boundary that is not also a picture boundary may result in propagation of mismatch error. An encoder can avoid such potential error propagation by avoiding the use of motion vectors that cause such references.

When more than one inter-layer constrained tile sets SEI message is present within the access units of a CVS, they shall contain identical content.

The number of inter-layer constrained tile sets SEI messages in each access unit shall not exceed 5.

il_all_tiles_exact_sample_value_match_flag equal to equal to 1 indicates that, within the CVS, when the coding tree blocks that are outside of any identified tile are not decoded and the boundaries of the identified tile is treated as picture boundaries for purposes of the decoding process, the value of each sample in the identified tile would be exactly the same as the value of the sample that would be obtained when all the coding tree blocks of all pictures in the CVS are decoded. il_all_tiles_exact_sample_value_match_flag equal to 0 indicates that, within the CVS, when the coding tree blocks that are outside of any identified tile are not decoded and the boundaries of the identified tile is treated as picture boundaries for purposes of the decoding process, the value of each sample in the identified tile may or may not be exactly the same as the value of the same sample when all the coding tree blocks of all pictures in the CVS are decoded.

il_one_tile_per_tile_set_flag equal to 1 indicates that each inter-layer constrained tile set contains one tile, and il_num_sets_in_message_minus1 is not present. When il_one_tile_per_tile_set_flag is equal to zero, tile sets are signalled explicitly.

il_num_sets_in_message_minus1 plus 1 specifies the number of inter-layer tile sets identified in the SEI message. The value of il_num_sets_in_message_minus1 shall be in the range of 0 to 255, inclusive.

skipped_tile_set_present_flag equal to 1 indicates that, within the CVS, the tile set consists of those remaining tiles that are not included in any earlier tile sets in the same message and all the prediction blocks that are inside the identified tile set having nuh_layer_id equal to ictsNuhLayerId are inter-layer predicted from inter-layer reference pictures with nuh_layer_id equal to RefLayerId[ictsNuhLayerId][NumDirectRefLayers[ictsNuhLayerId] – 1] and no residual_coding syntax structure is present in any transform unit of the identified tile set, where ictsNuhLayerId is the value of nuh_layer_id of this message. skipped_tile_set_present_flag equal to 0 does not indicate a bitstream constraint within the CVS. When not present, the value of skipped_tile_set_present_flag is inferred to be equal to 0.

ilcts_id[i] contains an identifying number that may be used to identify the purpose of the i-th identified tile set (for example, to identify an area to be extracted from the coded video sequence for a particular purpose). The value of ilcts_id[i] shall be in the range of 0 to $2^{32} - 2$, inclusive.

Values of ilcts_id[i] from 0 to 255 and from 512 to $2^{31} - 1$ may be used as determined by the application. Values of ilcts_id[i] from 256 to 511 and from 2^{31} to $2^{32} - 2$ are reserved for future use by ITU-T | ISO/IEC. Decoders encountering a value of ilcts_id[i] in the range of 256 to 511 or in the range of 2^{31} to $2^{32} - 2$ shall ignore (remove from the bitstream and discard) it.

il_num_tile_rects_in_set_minus1[i] plus 1 specifies the number of rectangular regions of tiles in the i-th identified inter-layer constrained tile set. The value of il_num_tile_rects_in_set_minus1[i] shall be in the range of 0 to (num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1) − 1, inclusive.

il_top_left_tile_index[i][j] and **il_bottom_right_tile_index**[i][j] identify the tile position of the top-left tile and the tile position of the bottom-right tile in a rectangular region of the i-th identified inter-layer constrained tile set, respectively, in tile raster scan order.

il_exact_sample_value_match_flag[i] equal to 1 indicates that, within the CVS, when the coding tree blocks that do not belong to the inter-layer constrained tile set are not decoded and the boundaries of the inter-layer constrained tile set are treated as picture boundaries for purposes of the decoding process, the value of each sample in the inter-layer constrained tile set would be exactly the same as the value of the sample that would be obtained when all the coding tree blocks of all pictures in the coded video sequence are decoded. il_exact_sample_value_match_flag[i] equal to 0 indicates that, within the CVS, when the coding tree blocks that are outside of the i-th identified inter-layer constrained tile set are not decoded and the boundaries of the inter-layer constrained tile set are treated as picture boundaries for purposes of the decoding process, the value of each sample in the identified tile set may or may not be exactly the same as the value of the same sample when all the coding tree blocks of the picture are decoded.

NOTE 2 – It should be feasible to use il_exact_sample_value_match_flag equal to 1 when using certain combinations of loop_filter_across_tiles_enabled_flag, pps_loop_filter_across_slices_enabled_flag, pps_deblocking_filter_disabled_flag, slice_loop_filter_across_slices_enabled_flag, slice_deblocking_filter_disabled_flag, sample_adaptive_offset_enabled_flag, slice_sao_luma_flag, and slice_sao_chroma_flag.

ilc_idc[i] equal to 1 indicates that, within the CVS, no samples outside of the i-th identified tile set and no samples at a fractional sample position that is derived using one or more samples outside of the i-th identified tile set are used for inter-layer prediction of any sample within the i-th identified tile set with nuh_layer_id equal to ictsNuhLayerId, where ictsNuhLayerId is the value of nuh_layer_id of this message. ilc_idc[i][j] equal to 2 indicates that, within the CVS, no prediction block in the i-th identified tile set with nuh_layer_id equal to ictsNuhLayerId is predicted from an inter-layer reference picture. ilc_idc[i] equal to 0 indicates that, within the CVS, the inter-layer prediction process may or may not be constrained for the prediction block in the i-th identified tile set having nuh_layer_id equal to ictsNuhLayerId. The value of ilc_idc[i] equal to 3 is reserved.

all_tiles_ilc_idc equal to 1 indicates that, within the CVS, no sample value outside of each identified tile and no sample value at a fractional sample position that is derived using one or more samples outside of the identified tile is used for inter-layer prediction of any sample within the identified tile with nuh_layer_id equal to ictsNuhLayerId, where ictsNuhLayerId is the value of nuh_layer_id of this message. all_tiles_ilc_idc equal to 2 indicates that, within the CVS, no prediction block in each identified tile with nuh_layer_id equal to ictsNuhLayerId is predicted from an inter-layer reference picture. all_tiles_ilc_idc equal to 0 indicates that, within the CVS, the inter-layer prediction process may or may not be constrained for the tile having nuh_layer_id equal to ictsNuhLayerId. The value of all_tiles_ilc_idc equal to 3 is reserved.

F.14.2.3 Bitstream partition nesting SEI message semantics

The bitstream partition nesting SEI message provides a mechanism to associate SEI messages with a bitstream partition of a layer set.

When present, this SEI message shall be contained within a scalable nesting SEI message. When this SEI message is contained in a scalable nesting SEI message, it shall be the only nested SEI message. In the scalable nesting SEI message containing this SEI message bitstream_subset_flag shall be equal to 1, nesting_op_flag is equal to 1, default_op_flag shall be equal to 0 and nesting_num_ops_minus1 shall be equal to 0. The nuh_layer_id of the SEI NAL unit shall be equal to the highest value within the list nestingLayerIdList[0].

A bitstream partition nesting SEI message contains one or more SEI messages.

bsp_idx specifies the bitstream partition index to which the contained SEI message apply as follows:

– If vps_vui_bsp_hrd_present_flag is equal to 1, bsp_idx is an index among the bitstream partitions specified for the layer set with index nesting_op_idx[0] in the vps_vui_bsp_hrd_parameters( ) syntax structure.

– Otherwise, an associated bitstream partition HRD parameters SEI message shall be present. The associated bitstream partition HRD parameter SEI message for the bitstream partition nesting SEI message is the preceding bitstream partition HRD parameters SEI message, in decoding order, that is nested in a scalable nesting SEI message with nesting_op_idx[i] that, with any value of i in the range of 0 to nesting_num_ops_minus1 of the scalable nesting SEI message containing the bitstream partition HRD parameters SEI message, is equal to nesting_op_idx[0] of the scalable nesting SEI message containing the bitstream partition nesting SEI message. It is a requirement of bitstream conformance that when bitstream partition nesting SEI message is present, it shall have an associated bitstream partition HRD message within the same coded video sequence. bsp_idx is an index among the bitstream partitions specified in the associated bitstream partition HRD parameters SEI message.

F.14.2.4 Bitstream partition initial arrival time SEI message semantics

The bitstream partition initial arrival time SEI message specifies the initial arrival times to be used in the bitstream-partition-specific CPB operation.

When present, this SEI message shall be contained within bitstream partition nesting SEI message that is contained in a scalable nesting SEI message. The same bitstream partition SEI message shall also contain a buffering period SEI message.

nal_initial_arrival_delay[i] specifies the initial arrival time for the i-th schedule combination of the bitstream partition to which this SEI message applies, when NAL HRD parameters are in use.

vcl_initial_arrival_delay[i] specifies the initial arrival time for the i-th schedule combination of the bitstream partition to which this SEI message applies, when VCL HRD parameters are in use.

F.14.2.5 Bitstream partition HRD parameters SEI message semantics

The bitstream partition HRD parameters SEI message specifies HRD parameters for bitstream-partition-specific CPB operation.

When present, this SEI message shall be contained within a scalable nesting SEI message in an initial IRAP access unit. When this SEI message is contained in a scalable nesting SEI message, it shall be the only nested SEI message. In the scalable nesting SEI message containing this SEI message, bitstream_subset_flag shall be equal to 1, nesting_op_flag shall be equal to 1 and default_op_flag shall be equal to 0. The nuh_layer_id of the SEI NAL unit shall be equal to the highest value within the lists nestingLayerIdList[h] with h in the range of 0 to nesting_num_ops_minus1, inclusive.

sei_num_bsp_hrd_parameters_minus1 plus 1 specifies the number of hrd_parameters( ) syntax structures present within this SEI message.

sei_bsp_cprms_present_flag[i] equal to 1 specifies that the HRD parameters that are common for all sub-layers are present in the i-th hrd_parameters( ) syntax structure in this SEI message. sei_bsp_cprms_present_flag[i] equal to 0 specifies that the HRD parameters that are common for all sub-layers are not present in the i-th hrd_parameters( ) syntax structure in this SEI message and are derived to be the same as the (i − 1)-th hrd_parameters( ) syntax structure in the in this SEI message. sei_bsp_cprms_present_flag[0] is inferred to be equal to 1.

For the subsequent syntax elements of this SEI message, the variable lsIdx is set equal to nesting_op_idx[h].

sei_num_bitstream_partitions_minus1[lsIdx] plus 1 specifies the number of bitstream partitions for which HRD parameters are specified for the layer set with index nesting_op_idx[h].

sei_layer_in_bsp_flag[lsIdx][i][j] specifies that the layer with index j is a part of bitstream partition with index i within the layer set with index lsIdx.

It is a requirement of bitstream conformance that bitstream partition with index j shall not include direct or indirect reference layers of any layers in bitstream partition i for any values of i and j in the range of 0 to sei_num_bitstream_partitions_minus1[h], inclusive, such that i is less than j.

sei_num_bsp_sched_combinations_minus1[lsIdx] plus 1 specifies the number of combinations of delivery schedules and hrd_parameters( ) specified for bitstream partitions for the layer set with index lsIdx.

sei_bsp_comb_hrd_idx[lsIdx][i][j] specifies the index of hrd_parameters( ) within this SEI message used in the i-th combination of a delivery schedule and hrd_parameters( ) specified for the bitstream partition with index j and for the layer set with index lsIdx.

sei_bsp_comb_sched_idx[lsIdx][i][j] specifies the index of a delivery schedule within the hrd_parameters( ) syntax structure with the index sei_bsp_comb_hrd_idx[lsIdx][i][j] that is used in the i-th combination of a delivery schedule and hrd_parameters( ) specified for the bitstream partition with index j and for the layer set with index lsIdx.

F.15 Video usability information

F.15.1 General

The specifications in clause E.1 apply.

F.15.2 VUI syntax

The specifications in clause E.2 apply.

F.15.3 VUI semantics

F.15.3.1 VUI parameters semantics

The specifications in clause E.3.1 apply with the following modifications and additions.

vui_timing_info_present_flag equal to 1 specifies that vui_num_units_in_tick, vui_time_scale, vui_poc_proportional_to_timing_flag, and vui_hrd_parameters_present_flag are present in the vui_parameters( ) syntax structure. vui_timing_info_present_flag equal to 0 specifies that vui_num_units_in_tick, vui_time_scale, vui_poc_proportional_to_timing_flag, and vui_hrd_parameters_present_flag are not present in the vui_parameters( ) syntax structure. It is a requirement of bitstream conformance that, when nuh_layer_id is greater than 0, vui_timing_info_present_flag shall be equal to 0.

F.15.3.2 HRD parameters semantics

The specifications in clause E.3.2 apply.

F.15.3.3 Sub-layer HRD parameters semantics

The specifications in clause 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 F.8.5, 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

Outputs of this process are updated lists of inter-layer pictures RefPicSetInterLayer0 and RefPicSetInterLayer1 and the variables NumActiveRefLayerPics0 and NumActiveRefLayerPics1.

The lists RefPicSetInterLayer0 and RefPicSetInterLayer1 are first emptied, NumActiveRefLayerPics0 and NumActiveRefLayerPics1 are set equal to 0 and the following applies:

```
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 ] ) {
[Ed. (CY): The rationale of allocating the split reference layer subsets based on the base view position seems to be unclear.]
        if( ( ViewId[ nuh_layer_id ]  <=  ViewId[ 0 ]   &&
                                    ViewId[ nuh_layer_id ]  <=  ViewId[ RefPicLayerId[ i ] ] )  | |
            ( ViewId[ nuh_layer_id ]  >=  ViewId[ 0 ]  &&
                                    ViewId[ nuh_layer_id ]  >=  ViewId[ RefPicLayerId[ i ] ] ) ) {
            RefPicSetInterLayer0[ NumActiveRefLayerPics0 ] = picX
            RefPicSetInterLayer0[ NumActiveRefLayerPics0++ ] is marked as "used for long-term reference"
        } else {
            RefPicSetInterLayer1[ NumActiveRefLayerPics1 ] = picX
            RefPicSetInterLayer1[ NumActiveRefLayerPics1++ ] is marked as "used for long-term reference"
        }
    } else
        RefPicSetInterLayer0[ NumActiveRefLayerPics0++ ] = "no reference picture"
}
```

There shall be no entry equal to "no reference picture" in RefPicSetInterLayer0 or RefPicSetInterLayer1.

If the current picture is a RADL picture, there shall be no entry in RefPicSetInterLayer0 or RefPicSetInterLayer1 that is a RASL picture.

NOTE – 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 < NumActiveRefLayerPics0; i++ )
    RefPicSetInterLayer0[ i ] is marked as "used for short-term reference"

for( i = 0; i <  NumActiveRefLayerPics1; i++ )
    RefPicSetInterLayer1[ 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 F.8.3.1 apply.

G.8.3.2 Decoding process for reference picture set

The specifications in subclause F.8.3.2 apply.

G.8.3.3 Decoding process for generating unavailable reference pictures

The specifications in subclause F.8.3.3 apply.

G.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, NumPicTotalCurr) and the list RefPicListTemp0 is constructed as follows:

```
rIdx = 0
while( rIdx < NumRpsCurrTempList0 ) {
    for( i = 0; i < NumPocStCurrBefore  &&  rIdx < NumRpsCurrTempList0; rIdx++, i++ )
        RefPicListTemp0[ rIdx ] = RefPicSetStCurrBefore[ i ]
    for( i = 0; i < NumActiveRefLayerPics0; rIdx++, i++ )
        RefPicListTemp0[ rIdx ] = RefPicSetInterLayer0[ i ]
    for( i = 0;  i < NumPocStCurrAfter  &&  rIdx < NumRpsCurrTempList0; rIdx++, i++ )          (G-1)
        RefPicListTemp0[ rIdx ] = RefPicSetStCurrAfter[ i ]
    for( i = 0; i < NumPocLtCurr  &&  rIdx < NumRpsCurrTempList0; rIdx++, i++ )
        RefPicListTemp0[ rIdx ] = RefPicSetLtCurr[ i ]
    for( i = 0; i < NumActiveRefLayerPics1; rIdx++, i++ )
        RefPicListTemp0[ rIdx ] = RefPicSetInterLayer1[ i ]
}
```

The list RefPicList0 is constructed as follows:

```
for( rIdx = 0; rIdx  <=  num_ref_idx_l0_active_minus1; rIdx++)                                (G-2)
    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, NumPicTotalCurr) and the list RefPicListTemp1 is constructed as follows:

```
rIdx = 0
while( rIdx < NumRpsCurrTempList1 ) {
    for( i = 0; i < NumPocStCurrAfter  &&  rIdx < NumRpsCurrTempList1; rIdx++, i++ )
        RefPicListTemp1[ rIdx ] = RefPicSetStCurrAfter[ i ]
    for( i = 0; i< NumActiveRefLayerPics1; rIdx++, i++ )
        RefPicListTemp1[ rIdx ] = RefPicSetInterLayer1[ i ]
    for( i = 0;  i < NumPocStCurrBefore  &&  rIdx < NumRpsCurrTempList1; rIdx++, i++ )         (G-3)
        RefPicListTemp1[ rIdx ] = RefPicSetStCurrBefore[ i ]
    for( i = 0; i < NumPocLtCurr  &&  rIdx < NumRpsCurrTempList1; rIdx++, i++ )
        RefPicListTemp1[ rIdx ] = RefPicSetLtCurr[ i ]
    for( i = 0; i< NumActiveRefLayerPics0; rIdx++, i++ )
        RefPicListTemp1[ rIdx ] = RefPicSetInterLayer0[ 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++)                                  (G-4)
    RefPicList1[ rIdx ] = ref_pic_list_modification_flag_l1 ? RefPicListTemp1[ list_entry_l1[ rIdx ] ] :
                                                              RefPicListTemp1[ rIdx ]
```

G.8.3.5 Decoding process for collocated picture and no backward prediction flag

The specifications in subclause F.8.3.5 apply.

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

The specifications in subclause F.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 F.8.6 apply.

G.8.7 In-loop filter process

The specifications in subclause F.8.7 apply.

G.9 Parsing process

The specifications in subclause F.9 apply.

G.10 Specification of bitstream subsets

The specifications in subclause F.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_multilayer_extension( ) syntax structure of the active SPS RBSP for the layer with nuh_layer_id equal to i.

– When ViewScalExtLayerFlag[i] is equal to 1, ScalabilityId[LayerIdxInVps [i]][smIdx] shall be equal to 0 for any smIdx value from 0 to 15, inclusive, that is not equal to 1, for any coded picture with nuh_layer_id equal to i.

– When ViewScalExtLayerFlag[i] is equal to 1, num_scaled_ref_layer_offsets shall be equal to 0 in each active SPS for the layer with nuh_layer_id equal to i.

– SPSs shall have sps_shvc_reserved_zero_idc equal to 0 only. [Ed. (GT): Corresponds to num_scaled_ref_layer_offsets in SHVC draft. Redundant with current constraint on sps_shvc_reserved_zero_idc.]

– When ViewScalExtLayerFlag[i] is equal to 1, the values of pic_width_in_luma_samples and pic_height_in_luma_samples in the active SPS for the layer with nuh_layer_id equal to i shall be equal to the values of pic_width_in_luma_samples and pic_height_in_luma_samples, respectively, in the active SPSs for all direct reference layers of that layer.

– 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 any smIdx from 0 to 15, inclusive that is not equal to 1.

– VPSs shall have avc_base_layer_flag equal to 0 only.

– VPSs shall have vps_num_rep_formats_minus1 in the range of 0 to 15, inclusive.

– SPSs shall have sps_extension_type_flag[i] equal to 0 only for i equal to 0, and in the range of 2 to 6, inclusive.

G.11.2 Tiers and levels

G.11.2.1 General tier and level limits

The specifications in A.4.1 apply with the following modifications.

[Ed: PicSizeInSamplesY corresponds to the spatial resolution of a picture; it is assumed that the picture size in each view is the same.]

[Ed: The current design assumes only two views are present.]

Replace item d) by the following:

d) The value of sps_max_dec_pic_buffering_minus1[HighestTid] + 1 shall be less than or equal to MaxDpbSize, which is derived as follows:

```
if( PicSizeInSamplesY  <=  ( MaxLumaPs  >>  2 ) )
      MaxDpbSize = Min( 4 * maxDpbPicBuf, 16 )
else if( PicSizeInSamplesY  <=  ( MaxLumaPs  >>  1 ) )
      MaxDpbSize = Min( 2 * maxDpbPicBuf, 16 )                                  (G-5)
else if( PicSizeInSamplesY  <=  ( ( 3 * MaxLumaPs )  >>  2 ) )
      MaxDpbSize = Min( ( 4 * maxDpbPicBuf ) / 3, 16 )
else
      MaxDpbSize = maxDpbPicBuf
```

where MaxLumaPs is specified in Table A-1 and maxDpbPicBuf is equal to 6.

[Ed: The above needs to be considered depending on the outcome of the DPB spec. If DPB is specified per layer, then no change is needed, it just needs to be clarified that these constraints apply per layer. But if the DPB is for all views, then certain parameters (e.g., MaxLumaPs and maxDpbBuf) should be doubled.]

G.11.2.2 Profile-specific level limits for the Main, Main 10, and Stereo Main profiles

[Ed (CY/JB): Currently the same level value may be intended to be used for a given resolution for both Stereo Main profile and Main profile, even though the stereo bitstream requires twice of the decoding capability as the single-view bitstream. Further study is needed to consider if this is an appropriate intention, and how to express the intention with appropriate constraints.]

The specifications in A.4.1 apply with the following modifications.

Replace item b) by the following:

b) The difference between consecutive output times of pictures of the same layer from the DPB, as specified in subclause C.3.3, shall satisfy the constraint that DpbOutputInterval[n] is greater than or equal to Max(PicSizeInSamplesY ÷ MaxLumaSr, fR) for the value of PicSizeInSamplesY of picture n, where MaxLumaSr is the value specified in Table A-2 for picture n, provided that picture n is a picture that is output and is not the last picture of the bitstream that is output.

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_flag[i])	
num_sample_shift_plus512[i]	u(10)
}	
three_dimensional_reference_displays_extension_flag	u(1)
}	

G.14.1.2 Depth representation information SEI message syntax

depth_representation_info_sei(payloadSize) {	**Descriptor**
z_near_flag	u(1)
z_far_flag	u(1)
d_min_flag	u(1)
d_max_flag	u(1)
depth_representation_type	ue(v)
if(d_min_flag || d_max_flag)	
disparity_ref_view_id	ue(v)
if(z_near_flag)	
depth_rep_sei_element(ZNearSign, ZNearExp, ZNearMantissa, ZNearManLen)	
if(z_far_flag)	
depth_rep_sei_element(ZFarSign, ZFarExp, ZFarMantissa, ZFarManLen)	
if(d_min_flag)	
depth_rep_sei_element(DMinSign, DMinExp, DMinMantissa, DMinManLen)	
if(d_max_flag)	
depth_rep_sei_element(DMaxSign, DMaxExp, DMaxMantissa, DMaxManLen)	
if(depth_representation_type = = 3) {	
depth_nonlinear_representation_num_minus1	ue(v)
for(i = 1; i <= depth_nonlinear_representation_num_minus1 + 1; i++)	
depth_nonlinear_representation_model[i]	

}	
}	

G.14.1.2.1 Depth representation SEI element syntax

depth_rep_sei_element(OutSign, OutExp, OutMantissa, OutManLen) {	**Descriptor**
da_sign_flag	u(1)
da_exponent	u(7)
da_mantissa_len_minus1	u(5)
da_mantissa	u(v)
}	

G.14.2 SEI message semantics

Table G-1 – Persistence scope of SEI messages (informative)

SEI message	Persistence scope
3D reference displays information	The access unit containing the SEI message and up to but not including the next access unit, in both decoding and displaying order, that contains the SEI message
Depth representation information	Specified by the semantics of the SEI message.

G.14.2.1 3D reference displays information SEI message semantics

A 3D reference displays information SEI message contains information about the reference display width(s) and reference viewing distance(s) as well as information about the corresponding reference stereo-pair(s) i.e. the pair(s) of views to be displayed for the viewer's left and right eyes on the reference display at the reference viewing distance. This information enables a view renderer to generate 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 pair of view specified in this SEI message can be used to extract or infer parameters related to the distance between the camera centers in the reference stereo-pair, which can be used for generation of views for the target display. For multi-view displays, the reference stereo-pair corresponds to a pair of views that can be simultaneously observed by the viewer's left and right eyes.

When present, this SEI message shall be associated an IRAP access unit or with a non-IRAP access unit, when all access units that follow this access unit in the decoding order, also follow it in the display order. The 3D reference display information SEI message should be applied for the access unit, it is associated with and the access units which follow this access unit in both the output and decoding order until the next IRAP access unit or the next access unit containing a 3D reference displays information SEI message.

NOTE 1 – The 3D 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 difference between the values of ViewId is referred to as the baseline distance (i.e. the distance between the centers of the cameras used to obtain the video sequence).

The following equations can be used for determining 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[i] = refBaseline[i] * (refDisplayWidth[i] ÷ displayWidth)

shift[i] = refShift[i] * (refDisplayWidth[i] ÷ displayWidth)

where refBaseline[i] is equal to right_view_id[i] − left_view_id[i] signalled in this SEI message. Other parameters related to the view generation may be obtained determined by using a similar equation.

parameter[i] = refParameter[i] * (refDisplayWidth[i] ÷ displayWidth)

where refParameter[i] is a parameter related to view generation that corresponds to the reference pair of views signalled by left_view_id[i] and right_view_id[i]. In the above 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 determining the pair of views and horizontal shift and/or other view synthesis parameters 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 above equations would be to keep the perceived depth in the same proportion to the viewing distance as in the reference setup.

When the view synthesis related parameters that correspond to the reference stereo-pair change from one access unit to another, they should be scaled with the same scaling factor as the parameters in the access unit that the SEI message is associated with. Therefore, the above equation should also be applied to obtain the parameters for a following access unit, where the refParameter is the parameter related to the reference stereo-pair associated the following access unit.

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).

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

right_view_id[i] indicates 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-2 are derived as follows from the respective variables or values in the e, n, and v rows of Table G-2 as follows:

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

$$x = 2^{(e-31)} * (1 + n \div 2^{v}) \qquad \text{(G-6)}$$

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

$$x = 2^{-(30+v)} * n \qquad \text{(G-7)}$$

NOTE 2 – The above specification is similar to that found in IEC 60559:1989, *Binary floating-point arithmetic for microprocessor systems*.

Table G-2 – 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 3 – 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.14.2.2 Depth representation information SEI message semantics

The syntax elements in the depth representation information SEI message specify various parameters for auxiliary pictures of type AUX_DEPTH for the purpose of processing decoded primary and auxiliary pictures prior to rendering on a 3D display, such as view synthesis. Specifically, depth or disparity ranges for depth pictures are specified.

When present, the depth representation information SEI message shall be associated with one or more layers with AuxId value equal to AUX_DEPTH. When the depth representation SEI message is not nested in the scalable nesting SEI message, it is associated with the layer having nuh_layer_id value equal to that of the SEI NAL unit containing the SEI message. When the depth representation SEI message is nested in the scalable nesting SEI message, it is associated with the layers having nuh_layer_id value equal to each value in the list nestingLayerIdList[i] for each value of i specified in the scalable nesting SEI message.

When present, the depth representation information SEI message may be included in any access unit. It is recommended that, when present, the SEI message is included in an IRAP access unit for the purpose of random access. The information indicated in the SEI message applies to all the pictures of each associated layer from the access unit containing the SEI message to the next access unit, in decoding order, containing an SEI message of the same type and associated with the same layer, exclusive, or to the end of the coded video sequence, whichever is earlier in decoding order.

z_near_flag equal to 0 specifies that the syntax elements specifying the nearest depth value are not present in the syntax structure. z_near_flag equal to 1 specifies that the syntax elements specifying the nearest depth value are present in the syntax structure.

z_far_flag equal to 0 specifies that the syntax elements specifying the farthest depth value are not present in the syntax structure. z_far_flag equal to 1 specifies that the syntax elements specifying the farthest depth value are present in the syntax structure.

d_min_flag equal to 0 specifies that the syntax elements specifying the minimum disparity value are not present in the syntax structure. d_min_flag equal to 1 specifies that the syntax elements specifying the minimum disparity value are present in the syntax structure.

d_max_flag equal to 0 specifies that the syntax elements specifying the maximum disparity value are not present in the syntax structure. d_max_flag equal to 1 specifies that the syntax elements specifying the maximum disparity value are present in the syntax structure.

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

[Ed. (MH): the semantics should be generalized to apply for other bit-depths than 8 or a constraint should be added that the luma bit-depth for the depth auxiliary pictures shall be equal to 8.]

Table G-3 – Definition of depth_representation_type

depth_representation_type	**Interpretation**
0	Each decoded luma sample value of an auxiliary picture represents an inverse of Z value that is uniformly quantized into the range of 0 to 255, inclusive.
1	Each decoded luma sample value of an auxiliary picture represents disparity that is uniformly quantized into the range of 0 to 255, inclusive.
2	Each decoded luma sample value of an auxiliary picture represents a Z value uniformly quantized into the range of 0 to 255, inclusive.

3	Each decoded luma sample value of an auxiliary picture represents a nonlinearly mapped disparity, normalized in range from 0 to 255, as specified by depth_nonlinear_representation_num_minus1 and depth_nonlinear_representation_model[i].
Other values	Reserved for future use

disparity_ref_view_id specifies the ViewId value against which the disparity values are derived.

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

– If 0 < e < 127, $x = (-1)^{s} * 2^{e-31} * (1 + n \div 2^{v})$.

– Otherwise (e is equal to 0), $x = (-1)^{s} * 2^{-(30+v)} * n$.

NOTE 1 – The above specification is similar to that found in IEC 60559:1989, *Binary floating-point arithmetic for microprocessor systems*.

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

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

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

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

depth_nonlinear_representation_num_minus1 plus 2 specifies the number of piecewise linear segments for mapping of depth values to a scale that is uniformly quantized in terms of disparity.

depth_nonlinear_representation_model[i] specifies the piecewise linear segments for mapping of decoded luma sample values of an auxiliary picture to a scale that is uniformly quantized in terms of disparity.

NOTE 2 – When depth_representation_type is equal to 3, an auxiliary picture contains nonlinearly transformed depth samples. Variable DepthLUT[i], as specified below, is used to transform coded depth sample values from nonlinear representation to the linear representation – disparity normalized in range from 0 to 255. The shape of this transform is defined by means of line-segment-approximation in two-dimensional linear-disparity-to-nonlinear-disparity space. The first (0, 0) and the last (255, 255) nodes of the curve are predefined. Positions of additional nodes are transmitted in form of deviations (depth_nonlinear_representation_model[i]) from the straight-line curve. These deviations are uniformly distributed along the whole range of 0 to 255, inclusive, with spacing depending on the value of nonlinear_depth_representation_num_minus1.

Variable DepthLUT[i] for i in the range of 0 to 255, inclusive, is specified as follows.

```
depth_nonlinear_representation_model[ 0 ] = 0
depth_nonlinear_representation_model[depth_nonlinear_representation_num_minus1 + 2 ] = 0
for( k=0; k  <=  depth_nonlinear_representation_num_minus1 + 1; ++k ) {
    pos1 = ( 255 * k ) / (depth_nonlinear_representation_num_minus1 + 2 )
    dev1 = depth_nonlinear_representation_model[ k ]
    pos2 = ( 255 * ( k+1 ) ) / (depth_nonlinear_representation_num_minus1 + 2 ) )
    dev2 = depth_nonlinear_representation_model[ k+1 ]

    x1 = pos1 − dev1
    y1 = pos1 + dev1
    x2 = pos2 − dev2
    y2 = pos2 + dev2

    for( x = Max( x1, 0 ); x  <=  Min( x2, 255 ); ++x )
            DepthLUT[ x ] = Clip3( 0, 255, Round( ( ( x − x1 ) * ( y2 − y1 ) ) ÷ ( x2 − x1 ) + y1 ) )
}
```

When depth_representation_type is equal to 3, DepthLUT[dS] for all decoded luma sample values dS of an auxiliary picture in the range of 0 to 255, inclusive, represents disparity that is uniformly quantized into the range of 0 to 255, inclusive.

G.14.2.2.1 Depth representation SEI element semantics

The syntax structure specifies the value of an element in the depth representation information SEI message.

The syntax structure sets the values of the OutSign, OutExp, OutMantissa, and OutManLen variables that represent a floating-point value. When the syntax structure is included in another syntax structure, the variable names OutSign, OutExp, OutMantissa, and OutManLen are to be interpreted as being replaced by the variable names used when the syntax structure is included. [Ed. (GT): The concept of syntax structures having output variables seems to be new. We might consider allowing alternative concepts. E.g. allowing syntax structures to have an index and dot operations would help also in other parts of the spec. E.g. ZFarExp = depth_rep_sei_element( )[1].da_exponent.]

da_sign_flag equal to 0 indicates that the sign of the floating-point value is positive. da_sign_flag equal to 1 indicates that the sign is negative. The variable OutSign is set equal to da_sign_flag.

da_exponent specifies the exponent of the floating-point value. The value of da_exponent shall be in the range of 0 to $2^7 - 2$, inclusive. The value $2^7 - 1$ is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value $2^7 - 1$ as indicating an unspecified value. The variable OutExp is set equal to da_exponent.

da_mantissa_len_minus1 plus 1 specifies the number of bits in the da_mantissa syntax element. The value of da_mantissa_len_minus1 shall be in the range of 0 to 31, inclusive. The variable OutManLen is set equal to da_mantissa_len_minus + 1.

da_mantissa specifies the mantissa of the floating-point value. The variable OutMantissa is set equal to da_mantissa.

G.15 Video usability information

The specifications in clause F.15 apply.