# SCALABLE NESTED SEI MESSAGE HANDLING IN VIDEO SUB-BITSTREAM EXTRACTION PROCESS

## CROSS REFERENCE TO RELATED APPLICATIONS

**[001]** This application is based on International Patent Application No. PCT/US2021/033717, filed on May 21, 2021, which claims the priority to and benefits of U.S. Provisional Patent Application No. 63/029,308, filed on May 22, 2020. All the aforementioned patent applications are hereby incorporated by reference in their entireties.

## TECHNICAL FIELD

**[002]** This patent document relates to image and video coding and decoding.

## BACKGROUND

**[003]** Digital video accounts for the largest bandwidth use on the internet and other digital communication networks. As the number of connected user devices capable of receiving and displaying video increases, it is expected that the bandwidth demand for digital video usage will continue to grow.

## SUMMARY

**[004]** The present document discloses techniques that can be used by video encoders and decoders to perform video encoding or decoding.

**[005]** In one example aspect, a video processing method is disclosed. The method includes performing a conversion between a video comprising multiple layers and a bitstream of the video according to a rule, wherein the rule specifies a maximum allowed value of a temporal layer identifier value of a sub-bitstream that is extracted from the bitstream.

**[006]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video and a bitstream of the video according to a rule, wherein the rule defines network abstraction layer (NAL) units to be extracted from the bitstream during a

sub-bitstream extraction process to output a sub-bitstream, and wherein the rule specifies to derive the sub-bitstream based on whether a list of NAL unit header identifier values in an output layer set (OLS) with a target OLS index does not include all values of NAL unit header identifiers in all video coding layer (VCL) NAL units in the bitstream that is input to the sub-bitstream extraction process.

**[007]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video and a bitstream of the video according to a rule, wherein the rule defines network abstraction layer (NAL) units to be extracted from the bitstream during a sub-bitstream extraction process to output a sub-bitstream, and wherein the rule specifies, responsive to a payload type of a first supplemental enhancement information (SEI) message included in an SEI network abstraction layer (NAL) unit, to disallow the SEI NAL unit to contain an SEI message with a certain payload type.

**[008]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video and a bitstream of the video according to a format rule, wherein the format rule specifies that the bitstream includes a flag specifying whether one or more non-scalable nested supplemental enhancement information (SEI) messages with one or more particular payload types apply to all output layer sets referenced by a coding layer.

**[009]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video and a bitstream of the video according to a rule, wherein the rule specifies a sub-bitstream extraction process by which an output sub-bitstream is extracted from the bitstream without removing network abstraction layer (NAL) units of a particular type and having a particular NAL unit header identifier value, wherein the particular type includes an access unit delimiter (AUD) NAL unit.

**[0010]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies, in a process of sub-bitstream extraction, to remove network abstraction layer (NAL) units that include a scalable nesting supplementary enhancement information (SEI) message applied to layers that are not included in a target output layer set (OLS).

**[0011]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies, in a process of sub-bitstream extraction, a non-

scalable nested supplementary enhancement information (SEI) message is generated by extracting an SEI message that is scalably-nested from a scalable nesting SEI message based on a first flag indicating whether the SEI message applies to specific output layer sets (OLSs) and a second flag indicating whether the SEI message applies to all subpictures or only to specific subpictures.

**[0012]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies, in a process of sub-bitstream extraction, a non-scalable nested supplemental enhancement information (SEI) message is generated by extracting multiple scalable-nested SEI messages from a first SEI network abstraction layer (NAL) unit in a picture unit.

**[0013]** In another example aspect, a video processing method is disclosed. The method includes performing a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies a sub-bitstream extraction process to generate an output bitstream, wherein the rule specifies handling of one or more supplemental enhancement information (SEI) network abstraction layer (NAL) units during the sub-bitstream extraction process.

**[0014]** In yet another example aspect, a video encoder apparatus is disclosed. The video encoder comprises a processor configured to implement above-described methods.

**[0015]** In yet another example aspect, a video decoder apparatus is disclosed. The video decoder comprises a processor configured to implement above-described methods.

**[0016]** In yet another example aspect, a computer readable medium having code stored thereon is disclose. The code embodies one of the methods described herein in the form of processor-executable code.

**[0017]** These, and other, features are described throughout the present document.

## BRIEF DESCRIPTION OF DRAWINGS

**[0018]** FIG. 1 is a block diagram that illustrates a video coding system in accordance with some embodiments of the disclosed technology.

**[0019]** FIG. 2 is a block diagram of an example hardware platform used for video processing.

**[0020]** FIG. 3 is a flowchart for an example method of video processing.

**[0021]** FIG. 4 is a block diagram that illustrates an example video coding system.

**[0022]** FIG. 5 is a block diagram that illustrates an encoder in accordance with some embodiments of the disclosed technology.

**[0023]** FIG. 6 is a block diagram that illustrates a decoder in accordance with some embodiments of the disclosed technology.

**[0024]** FIGS. 7A to 7D are flowcharts for example methods of video processing based on some implementations of the disclosed technology.

**[0025]** FIG. 8 is a flowchart for an example method of video processing based on some implementations of the disclosed technology.

**[0026]** FIG. 9 is a flowchart for an example method of video processing based on some implementations of the disclosed technology.

**[0027]** FIGS. 10A to 10C are flowcharts for example methods of video processing based on some implementations of the disclosed technology.

## DETAILED DESCRIPTION

**[0028]** Section headings are used in the present document for ease of understanding and do not limit the applicability of techniques and embodiments disclosed in each section only to that section. Furthermore, H.266 terminology is used in some description only for ease of understanding and not for limiting scope of the disclosed techniques. As such, the techniques described herein are applicable to other video codec protocols and designs also.

## 1. Introduction

**[0029]** This document is related to video coding technologies. Specifically, it is about some improvements on the general sub-bitstream extraction process, signalling of picture-level HRD parameters, and containing of SEI messages in SEI NAL units. The ideas may be applied individually or in various combination, to any video coding standard or non-standard video codec that supports multi-layer video coding, e.g., the being-developed Versatile Video Coding (VVC).

## 2. Abbreviations

| | |
|---|---|
| APS | Adaptation Parameter Set |
| AU | Access Unit |
| AUD | Access Unit Delimiter |

| | |
|---|---|
| AVC | Advanced Video Coding |
| CLVS | Coded Layer Video Sequence |
| CPB | Coded Picture Buffer |
| CRA | Clean Random Access |
| CTU | Coding Tree Unit |
| CVS | Coded Video Sequence |
| DCI | Decoding Capability Information |
| DPB | Decoded Picture Buffer |
| EOB | End Of Bitstream |
| EOS | End Of Sequence |
| GDR | Gradual Decoding Refresh |
| HEVC | High Efficiency Video Coding |
| HRD | Hypothetical Reference Decoder |
| IDR | Instantaneous Decoding Refresh |
| ILP | Inter-Layer Prediction |
| ILRP | Inter-Layer Reference Picture |
| JEM | Joint Exploration Model |
| LTRP | Long-Term Reference Picture |
| MCTS | Motion-Constrained Tile Sets |
| NAL | Network Abstraction Layer |
| OLS | Output Layer Set |
| PH | Picture Header |
| PPS | Picture Parameter Set |
| PTL | Profile, Tier and Level |
| PU | Picture Unit |
| RAP | Random Access Point |
| RBSP | Raw Byte Sequence Payload |
| SEI | Supplemental Enhancement Information |
| SPS | Sequence Parameter Set |
| STRP | Short-Term Reference Picture |
| SVC | Scalable Video Coding |

VCL Video Coding Layer
VPS Video Parameter Set
VTM VVC Test Model
VUI Video Usability Information
VVC Versatile Video Coding

## 3. Initial discussion

**[0030]** Video coding standards have evolved primarily through the development of the well-known ITU-T and ISO/IEC standards. The ITU-T produced H.261 and H.263, ISO/IEC produced MPEG-1 and MPEG-4 Visual, and the two organizations jointly produced the H.262/MPEG-2 Video and H.264/MPEG-4 Advanced Video Coding (AVC) and H.265/HEVC standards. Since H.262, the video coding standards are based on the hybrid video coding structure wherein temporal prediction plus transform coding are utilized. To explore the future video coding technologies beyond HEVC, the Joint Video Exploration Team (JVET) was founded by VCEG and MPEG jointly in 2015. Since then, many new methods have been adopted by JVET and put into the reference software named Joint Exploration Model (JEM). The JVET meeting is concurrently held once every quarter, and the new coding standard is targeting at 50% bitrate reduction as compared to HEVC. The new video coding standard was officially named as Versatile Video Coding (VVC) in the April 2018 JVET meeting, and the first version of VVC test model (VTM) was released at that time. As there are continuous effort contributing to VVC standardization, new coding techniques are being adopted to the VVC standard in every JVET meeting. The VVC working draft and test model VTM are then updated after every meeting. The VVC project is now aiming for technical completion (FDIS) at the July 2020 meeting.

### 3.1. Picture resolution change within a sequence

**[0031]** In AVC and HEVC, the spatial resolution of pictures cannot change unless a new sequence using a new SPS starts, with an IRAP picture. VVC enables picture resolution change within a sequence at a position without encoding an IRAP picture, which is always intra-coded. This feature is sometimes referred to as reference picture resampling (RPR), as the feature needs resampling of a reference picture used for inter prediction when that reference picture has a different resolution than the current picture being decoded.

**[0032]** The scaling ratio is restricted to be larger than or equal to 1/2 (2 times downsampling from the reference picture to the current picture), and less than or equal to 8 (8 times upsampling). Three

sets of resampling filters with different frequency cutoffs are specified to handle various scaling ratios between a reference picture and the current picture. The three sets of resampling filters are applied respectively for the scaling ratio ranging from 1/2 to 1/1.75, from 1/1.75 to 1/1.25, and from 1/1.25 to 8. Each set of resampling filters has 16 phases for luma and 32 phases for chroma which is same to the case of motion compensation interpolation filters. Actually the normal MC interpolation process is a special case of the resampling process with scaling ratio ranging from 1/1.25 to 8. The horizontal and vertical scaling ratios are derived based on picture width and height, and the left, right, top and bottom scaling offsets specified for the reference picture and the current picture.

**[0033]** Other aspects of the VVC design for support of this feature that are different from HEVC include: i) The picture resolution and the corresponding conformance window are signaled in the PPS instead of in the SPS, while in the SPS the maximum picture resolution is signaled. ii) For a single-layer bitstream, each picture store (a slot in the DPB for storage of one decoded picture) occupies the buffer size as required for storing a decoded picture having the maximum picture resolution.

## 3.2. Scalable video coding (SVC) in general and in VVC

**[0034]** Scalable video coding (SVC, sometimes also just referred to as scalability in video coding) refers to video coding in which a base layer (BL), sometimes referred to as a reference layer (RL), and one or more scalable enhancement layers (ELs) are used. In SVC, the base layer can carry video data with a base level of quality. The one or more enhancement layers can carry additional video data to support, for example, higher spatial, temporal, and/or signal-to-noise (SNR) levels. Enhancement layers may be defined relative to a previously encoded layer. For example, a bottom layer may serve as a BL, while a top layer may serve as an EL. Middle layers may serve as either ELs or RLs, or both. For example, a middle layer (e.g., a layer that is neither the lowest layer nor the highest layer) may be an EL for the layers below the middle layer, such as the base layer or any intervening enhancement layers, and at the same time serve as a RL for one or more enhancement layers above the middle layer. Similarly, in the Multiview or 3D extension of the HEVC standard, there may be multiple views, and information of one view may be utilized to code (e.g., encode or decode) the information of another view (e.g., motion estimation, motion vector prediction and/or other redundancies).

**[0035]** In SVC, the parameters used by the encoder or the decoder are grouped into parameter sets based on the coding level (e.g., video-level, sequence-level, picture-level, slice level, etc.) in which they may be utilized. For example, parameters that may be utilized by one or more coded video sequences of different layers in the bitstream may be included in a video parameter set (VPS), and parameters that are utilized by one or more pictures in a coded video sequence may be included in a sequence parameter set (SPS). Similarly, parameters that are utilized by one or more slices in a picture may be included in a picture parameter set (PPS), and other parameters that are specific to a single slice may be included in a slice header. Similarly, the indication of which parameter set(s) a particular layer is using at a given time may be provided at various coding levels.

**[0036]** Thanks to the support of reference picture resampling (RPR) in VVC, support of a bitstream containing multiple layers, e.g., two layers with SD and HD resolutions in VVC can be designed without the need any additional signal-processing-level coding tool, as upsampling needed for spatial scalability support can just use the RPR upsampling filter. Nevertheless, high-level syntax changes (compared to not supporting scalability) are needed for scalability support. Scalability support is specified in VVC version 1. Different from the scalability supports in any earlier video coding standards, including in extensions of AVC and HEVC, the design of VVC scalability has been made friendly to single-layer decoder designs as much as possible. The decoding capability for multi-layer bitstreams are specified in a manner as if there were only a single layer in the bitstream. E.g., the decoding capability, such as DPB size, is specified in a manner that is independent of the number of layers in the bitstream to be decoded. Basically, a decoder designed for single-layer bitstreams does not need much change to be able to decode multi-layer bitstreams. Compared to the designs of multi-layer extensions of AVC and HEVC, the HLS aspects have been significantly simplified at the sacrifice of some flexibilities. For example, an IRAP AU is required to contain a picture for each of the layers present in the CVS.

## 3.3. Parameter sets

**[0037]** AVC, HEVC, and VVC specify parameter sets. The types of parameter sets include SPS, PPS, APS, and VPS. SPS and PPS are supported in all of AVC, HEVC, and VVC. VPS was introduced since HEVC and is included in both HEVC and VVC. APS was not included in AVC or HEVC but is included in the latest VVC draft text.

**[0038]** SPS was designed to carry sequence-level header information, and PPS was designed to carry infrequently changing picture-level header information. With SPS and PPS, infrequently

changing information need not to be repeated for each sequence or picture, hence redundant signalling of this information can be avoided. Furthermore, the use of SPS and PPS enables out-of-band transmission of the important header information, thus not only avoiding the need for redundant transmissions but also improving error resilience.

**[0039]** VPS was introduced for carrying sequence-level header information that is common for all layers in multi-layer bitstreams.

**[0040]** APS was introduced for carrying such picture-level or slice-level information that needs quite some bits to code, can be shared by multiple pictures, and in a sequence there can be quite many different variations.

## 3.4. General sub-bitstream extraction process

Clause C.6 of the latest VVC text specifies a general sub-bitstream extraction process, as follows:

**C.6 Sub-bitstream extraction process**

Inputs to this process are a bitstream inBitstream, a target OLS index targetOlsIdx, and a target highest TemporalId value tIdTarget.

Output of this process is a sub-bitstream outBitstream.

It is a requirement of bitstream conformance for the input bitstream that any output sub-bitstream that satisfies all of the following conditions shall be a conforming bitstream:

- The output sub-bitstream is the output of the process specified in this clause with the bitstream, targetOlsIdx equal to an index to the list of OLSs specified by the VPS, and tIdTarget equal to any value in the range of 0 to 6, inclusive, as inputs.
- The output sub-bitstream contains at least one VCL NAL unit with nuh_layer_id equal to each of the nuh_layer_id values in LayerIdInOls[ targetOlsIdx ].
- The output sub-bitstream contains at least one VCL NAL unit with TemporalId equal to tIdTarget.

  NOTE – A conforming bitstream contains one or more coded slice NAL units with TemporalId equal to 0, but does not have to contain coded slice NAL units with nuh_layer_id equal to 0.

The output sub-bitstream OutBitstream is derived as follows:

1. The bitstream outBitstream is set to be identical to the bitstream inBitstream.
2. Remove from outBitstream all NAL units with TemporalId greater than tIdTarget.
3. Remove from outBitstream all NAL units with nal_unit_type not equal to any of VPS_NUT, DCI_NUT, and EOB_NUT and with nuh_layer_id not included in the list LayerIdInOls[ targetOlsIdx ].
4. Remove from outBitstream all VCL NAL units for which all of the following conditions are true, and their associated non-VCL NAL units with nal_unit_type equal to PH_NUT, FD_NUT, SUFFIX_SEI_NUT, and PREFIX_SEI_NUT with PayloadType not equal to 0, 1, or 130:
   - nal_unit_type is equal to TRAIL_NUT, STSA_NUT, RADL_NUT, or RASL_NUT, or nal_unit_type is equal to GDR_NUT and the associated ph_recovery_poc_cnt is not equal to 0.

- nuh_layer_id is equal to LayerIdInOls[ targetOlsIdx ][ j ] for a value of j in the range of 0 to NumLayersInOls[ targetOlsIdx ] − 1 inclusive.
- TemporalId is greater than or equal to NumSubLayersInLayerInOLS[ targetOlsIdx ][ GeneralLayerIdx[ nuh_layer_id ] ].

5. Remove from outBitstream all SEI NAL units that contain a scalable nesting SEI message that has sn_ols_flag equal to 1 and there is no value of i in the range of 0 to sn_num_olss_minus1, inclusive, such that NestingOlsIdx[ i ] is equal to targetOlsIdx.
6. When LayerIdInOls[ targetOlsIdx ] does not include all values of nuh_layer_id in all NAL units in the bitstream, the following applies:
   a. Remove from outBitstream all SEI NAL units that contain a non-scalable-nested SEI message with payloadType equal to 0 (BP) or 130 (DUI).
   b. When general_same_pic_timing_in_all_ols_flag is equal to 0, remove from outBitstream all SEI NAL units that contain a non-scalable-nested SEI message with payloadType equal to 1 (PT).
   c. When outBitstream contains SEI NAL units that contain a scalable nesting SEI message with sn_ols_flag equal to 1 and are applicable to outBitstream (NestingOlsIdx[ i ] is equal to targetOlsIdx), the following applies:
      - If general_same_pic_timing_in_all_ols_flag is equal to 0, extract appropriate non-scalable-nested SEI message with payloadType equal to 0 (BP), 1 (PT), or 130 (DUI) from the scalable nesting SEI message and include those SEI messages in outBitstream.
      - Otherwise (general_same_pic_timing_in_all_ols_flag is equal to 1), extract appropriate non-scalable-nested SEI message with payloadType equal to 0 (BP) or 130 (DUI) from the scalable nesting SEI message and include those SEI messages in outBitstream.

## 4. Technical problems solved by disclosed technical solutions

**[0041]** The existing designs of the general sub-bitstream extraction process and related other parts in the latest VVC text (in JVET-R2001-vA/v10) have the following problems:

1) In the conditions under which an output sub-bitstream is required to be a conforming bitstream, the value of tIdTarget is said to be in the range of 0 to 6, inclusive. However, in many bitstreams, the highest TemporalId value is less than 6, and that value is specified by the syntax element vps_max_sublayers_minus1.
2) An access unit delimiter (AUD) NAL unit, when present, can have any nuh_layer_id value. However, step 3 of the sub-bitstream extraction process would remove the AUD NAL units for which the nuh_layer_id values are not included in the list LayerIdInOls[ targetOlsIdx ].
3) Some SEI NAL units contain a scalable nesting SEI message with sn_ols_flag equal to 0 while the applicable layers as indicted in the scalable nesting SEI message does not included any layer in the target OLS, i.e., none of the applicable layers' nuh_layer_id values

is not included in the list LayerIdInOls[ targetOlsIdx ]. These SEI NAL units should also be removed.

4) The condition of step 6, i.e., "When LayerIdInOls[ targetOlsIdx ] does not include all values of nuh_layer_id in all NAL units in the bitstream" has the following two issues.
    a. The condition does not work for cases when DCI, VPS, AUD, or EOB NAL units are present and have nuh_layer_id not equal to any of the nuh_layer_id values of the VCL NAL units.
    b. The phrase "the bitstream" is not clear, as there are two bitstreams involved in the context, inBitstream and outBitstream.
5) Step 6.c would extract scalable-nested SEI messages, to generate non-scalable-nested SEI messages, from scalable nesting SEI messages with both sn_ols_flag equal to 1 and sn_subpic_flag equal to 1, while such scalable-nested SEI messages only apple to specific subpictures and thus should not be extracted.
6) In step 6.c, when multiple scalable-nested SEI messages are extracted from one SEI NAL unit seiNalUnitA to be non-scalable-nested SEI messages, they should still be included in one SEI NAL unit seiNalUnitB, and the SEI NAL unit seiNalUnitB should be included in the same PU that contained the SEI NAL unit seiNalUnitA. However, this is not specified.
7) Step 6.c should remove, from outBitstream, all SEI NAL units from which some SEI messages have been extracted and included as non-scalable-nested SEI messages. However, this is not specified.
8) There lacks a constraint such that when an SEI NAL unit contain an SEI message with payloadType equal to 0, 1, or 130, the SEI NAL unit shall not contain an SEI message with payloadType not equal to 0 (BP), 1 (PT), 130 (DUI), or 133 (scalable nesting). This causes the removal of SEI messages in step 4 involves more than just removal of SEI NAL units.
9) The flag general_same_pic_timing_in_all_ols_flag only specifies whether non-scalable-nested PT SEI messages apply to all OLSs. However, information carried in the DUI SEI messages are for similar purposes as in the PT SEI messages.

## 5. A listing of technical solutions and embodiments

**[0042]** To solve the above problems, and others, methods as summarized below are disclosed. The items should be considered as examples to explain the general concepts and should not be

interpreted in a narrow way. Furthermore, these items can be applied individually or combined in any manner.

1) To solve problem 1, the conditions under which an output sub-bitstream is required to be a conforming bitstream is specified such that the value of tIdTarget is specified be in the range of 0 to vps_max_sublayers_minus1, inclusive.
    a. Alternatively, the conditions under which an output sub-bitstream is required to be a conforming bitstream is specified such that the value of tIdTarget is specified be in the range of 0 to vps_max_sublayers_minus1, inclusive, when there is more than one layer in the input bitstream, and specified be in the range of 0 to sps_max_sublayers_minus1, inclusive, when there is only one layer in the input bitstream.
2) To solve problem 2, the general sub-bitstream extraction process is specified such that AUD NAL units are treated in the same manner as NAL units with nal_unit_type equal to VPS_NUT, DCI_NUT, or EOB_NUT. In other words, no AUD NAL unit is removed from the output bitstream outBitstream according to the nuh_layer_id value.
3) To solve problem 3, the general sub-bitstream extraction process is specified such that it would remove, the output bitstream outBitstream, SEI NAL units that contain a scalable nesting SEI message with sn_ols_flag equal to 0 while the applicable layers as indicted in the scalable nesting SEI message does not included any layer in the target OLS.
    a. In one example, it is specified to remove from outBitstream all SEI NAL units that contain a scalable nesting SEI message that has sn_ols_flag equal to 0 and there is no value in the list nestingLayerId[ i ] for i in the range of 0 to nestingNumLayers − 1, inclusive, that is in the list LayerIdInOls[ targetOlsIdx ].
4) To solve problem 4, the condition "When LayerIdInOls[ targetOlsIdx ] does not include all values of nuh_layer_id in all NAL units in the bitstream" is changed to be "When the list LayerIdInOls[ targetOlsIdx ] does not include all values of nuh_layer_id in all VCL NAL units in the bitstream inBitstream".
5) To solve problem 5, the general sub-bitstream extraction process is specified such that it only extracts scalable-nested SEI messages from scalable nesting SEI messages with both sn_ols_flag equal to 1 and sn_subpic_flag equal to 0 to generate non-scalable-nested SEI messages.

6) To solve problem 6, the general sub-bitstream extraction process is specified such that, when multiple scalable-nested SEI messages are extracted from one SEI NAL unit seiNalUnitA to be non-scalable-nested SEI messages, they are still included in one SEI NAL unit seiNalUnitB in the output bitstream outBitstream, and the SEI NAL unit seiNalUnitB is included in the PU that contained the SEI NAL unit seiNalUnitA.
7) To solve problem 7, the general sub-bitstream extraction process is specified such that it removes, from the output bitstream outBitstream, all SEI NAL units from which some SEI messages have been extracted and included as non-scalable-nested SEI messages.
    a. Alternatively, when the scalable-nested SEI messages in such an SEI NAL unit apply only to the target OLS (i.e., the targetOlsIdx-th OLS specified by the VPS), remove the SEI NAL unit from outBitstream.
    b. Alternatively, when there is no OLS, other than the target OLS in the OLSs to which the scalable-nested SEI messages in such an SEI NAL unit apply, that contains layers that are all included in the list LayerIdInOls[ targetOlsIdx ], remove the SEI NAL unit from outBitstream.
8) To solve problem 8, add constraint such that when an SEI NAL unit contain an SEI message with payloadType equal to 0, 1, or 130, the SEI NAL unit shall not contain an SEI message with payloadType not equal to 0 (BP), 1 (PT), 130 (DUI), or 133 (scalable nesting).
9) To solve problem 9, the flag general_same_pic_timing_in_all_ols_flag specifies whether non-scalable-nested PT and DUI SEI messages apply to all OLSs.
    a. Alternatively, the flag general_same_pic_timing_in_all_ols_flag specifies whether non-scalable-nested BP, PT, and DUI SEI messages apply to all OLSs.
        i. In one example, the flag general_same_pic_timing_in_all_ols_flag is renamed to be flag general_same_pic_level_hrd_info_in_all_ols_flag, which specifies whether non-scalable-nested BP, PT, and DUI SEI messages apply to all OLSs.
    b. Alternatively, a new flag, e.g., named general_same_dui_in_all_ols_flag, is added, to specify whether non-scalable-nested DUI SEI messages apply to all OLSs.
    c. Alternatively, a new flag, e.g., named general_same_bp_in_all_ols_flag, is added, to specify whether non-scalable-nested BP SEI messages apply to all OLSs.

# 6. Embodiments

*Below are some example embodiments for some of the invention aspects summarized above in Section 5, which can be applied to the VVC specification. The changed texts are based on the latest VVC text in JVET-R2001-vA/v10. Most relevant parts that have been added or modified are highlighted in* ***bold and Italic****, and some of the deleted parts are marked with double brackets (e.g., [[a]] denotes the deletion of the character "a"). There may be some other changes that are editorial in nature and thus not highlighted.*

## 6.1. First embodiment

This embodiment is for items 1, 2, 3, 3.a, 4, 5, 6, 7.b, and 8.

### C.6 *General* sub-bitstream extraction process

Inputs to this process are a bitstream inBitstream, a target OLS index targetOlsIdx, and a target highest TemporalId value tIdTarget.

Output of this process is a sub-bitstream outBitstream.

It is a requirement of bitstream conformance for the input bitstream that any output sub-bitstream that satisfies all of the following conditions shall be a conforming bitstream:

- The output sub-bitstream is the output of the process specified in this clause with the bitstream, targetOlsIdx equal to an index to the list of OLSs specified by the VPS, and tIdTarget equal to any value in the range of 0 to ***vps_max_sublayers_minus1***, inclusive, as inputs.
- The output sub-bitstream contains at least one VCL NAL unit with nuh_layer_id equal to each of the nuh_layer_id values in LayerIdInOls[ targetOlsIdx ].
- The output sub-bitstream contains at least one VCL NAL unit with TemporalId equal to tIdTarget.

NOTE – A conforming bitstream contains one or more coded slice NAL units with TemporalId equal to 0, but does not have to contain coded slice NAL units with nuh_layer_id equal to 0.

The output sub-bitstream OutBitstream is derived ***by applying the following ordered steps***:

1. The bitstream outBitstream is set to be identical to the bitstream inBitstream.
2. Remove from outBitstream all NAL units with TemporalId greater than tIdTarget.
3. Remove from outBitstream all NAL units with nal_unit_type not equal to any of DCI_NUT, VPS_NUT, ***AUD_NUT,*** and EOB_NUT and with nuh_layer_id not included in the list LayerIdInOls[ targetOlsIdx ].
4. Remove from outBitstream all VCL NAL units for which all of the following conditions are true, and ***also remove from outBitstream these VCL NAL units'*** associated non-VCL NAL units ***that have*** nal_unit_type equal to PH_NUT or FD_NUT, ***or have nal_unit_type equal to*** SUFFIX_SEI_NUT ***or*** PREFIX_SEI_NUT ***and contain SEI messages with payloadType*** not equal to 0 (BP), 1 (PT), 130 (DUI), ***or 133 (scalable nesting)***:
   - nal_unit_type is equal to TRAIL_NUT, STSA_NUT, RADL_NUT, or RASL_NUT, or nal_unit_type is equal to GDR_NUT and the associated ph_recovery_poc_cnt is not equal to 0.
   - [[ nuh_layer_id is equal to LayerIdInOls[ targetOlsIdx ][ j ] for a value of j in the range of 0 to NumLayersInOls[ targetOlsIdx ] − 1 inclusive. ]]

– TemporalId is greater than or equal to NumSubLayersInLayerInOLS[ targetOlsIdx ][ GeneralLayerIdx[ nuh_layer_id ] ].

5. Remove from outBitstream all SEI NAL units that contain a scalable nesting SEI message that has sn_ols_flag equal to 1 and there is no value of i in the range of 0 to sn_num_olss_minus1, inclusive, such that NestingOlsIdx[ i ] is equal to targetOlsIdx.
6. ***Remove from outBitstream all SEI NAL units that contain a scalable nesting SEI message that has sn_ols_flag equal to 0 and there is no value in the list nestingLayerId[ i ] for i in the range of 0 to nestingNumLayers − 1, inclusive, that is in the list LayerIdInOls[ targetOlsIdx ].***
7. When LayerIdInOls[ targetOlsIdx ] does not include all values of nuh_layer_id in all ***VCL*** NAL units in the bitstream ***inBitstream***, the following applies ***in the order listed***:
   a. Remove from outBitstream all SEI NAL units that contain a non-scalable-nested SEI message with payloadType equal to 0 (BP) or 130 (DUI).
   b. When general_same_pic_timing_in_all_ols_flag is equal to 0, remove from outBitstream all SEI NAL units that contain a non-scalable-nested SEI message with payloadType equal to 1 (PT).
   c. When outBitstream contains SEI NAL units that contain a scalable nesting SEI message with sn_ols_flag equal to 1 ***and sn_subpic_flag equal to 0*** that applies to the targetOlsIdx-th OLS (i.e., there is at least one value of i in the range of 0 to sn_num_olss_minus1, inclusive, such that NestingOlsIdx[ i ] is equal to targetOlsIdx), the following applies ***in the order listed***:
      i. For each scalable-nested BP or DUI SEI message in such an SEI NAL unit seiNalUnitA, ***generate a non-scalable-nested SEI message with the same payloadType and SEI payload and include it in an SEI NAL unit in the PU containing seiNalUnitA*** in outBitstream.
      ii. When general_same_pic_timing_in_all_ols_flag is equal to 0, for each scalable-nested PT SEI message in such an SEI NAL unit seiNalUnitA, ***generate a non-scalable-nested SEI message with the same SEI payload and include it in an SEI NAL unit in the PU containing seiNalUnitA*** in outBitstream.
      ***iii. When multiple SEI messages contained in a particular such SEI NAL unit seiNalUnitA were made non-scalable-nested SEI messages, thuse non-scalable-nested SEI messages are included in one SEI NAL unit.***
      ***iv. When there is no OLS, other than the target OLS in the OLSs to which the scalable-nested SEI messages in such an SEI NAL unit seiNalUnitA apply, that contains layers that are all included in the list LayerIdInOls[ targetOlsIdx ], remove the SEI NAL unit seiNalUnitA from outBitstream.***

**D.2.2 General SEI payload semantics**

...

It is a requirement of bitstream conformance that the following restrictions apply on containing of SEI messages in SEI NAL units:

- When an SEI NAL unit contains a non-scalable-nested BP SEI message, a non-scalable-nested PT SEI message, or a non-scalable-nested DUI SEI message, the SEI NAL unit shall not contain any other SEI message with payloadType not equal to 0 (BP), 1 (PT), or 130 (DUI).
- When an SEI NAL unit contains a scalable-nested BP SEI message, a scalable-nested PT SEI message, or a scalable-nested DUI SEI message, the SEI NAL unit shall not contain any other SEI message with payloadType not equal to 0 (BP), 1 (PT), 130 (DUI) or 133 (scalable nesting).
- ***When an SEI NAL unit contain an SEI message with payloadType equal to 0, 1, or 130, the SEI NAL unit shall not contain an SEI message with payloadType not equal to 0 (BP), 1 (PT), 130 (DUI), or 133 (scalable nesting).***

...

**[0043]** FIG. 1 is a block diagram showing an example video processing system 1900 in which various techniques disclosed herein may be implemented. Various implementations may include some or all of the components of the system 1900. The system 1900 may include input 1902 for receiving video content. The video content may be received in a raw or uncompressed format, e.g., 8 or 10 bit multi-component pixel values, or may be in a compressed or encoded format. The input 1902 may represent a network interface, a peripheral bus interface, or a storage interface. Examples of network interface include wired interfaces such as Ethernet, passive optical network (PON), etc. and wireless interfaces such as Wi-Fi or cellular interfaces.

**[0044]** The system 1900 may include a coding component 1904 that may implement the various coding or encoding methods described in the present document. The coding component 1904 may reduce the average bitrate of video from the input 1902 to the output of the coding component 1904 to produce a coded representation of the video. The coding techniques are therefore sometimes called video compression or video transcoding techniques. The output of the coding component 1904 may be either stored, or transmitted via a communication connected, as represented by the component 1906. The stored or communicated bitstream (or coded) representation of the video received at the input 1902 may be used by the component 1908 for generating pixel values or displayable video that is sent to a display interface 1910. The process of generating user-viewable video from the bitstream representation is sometimes called video decompression. Furthermore, while certain video processing operations are referred to as "coding" operations or tools, it will be appreciated that the coding tools or operations are used at an encoder and corresponding decoding tools or operations that reverse the results of the coding will be performed by a decoder.

**[0045]** Examples of a peripheral bus interface or a display interface may include universal serial bus (USB) or high definition multimedia interface (HDMI) or Displayport, and so on. Examples of storage interfaces include SATA (serial advanced technology attachment), PCI, IDE interface, and the like. The techniques described in the present document may be embodied in various electronic devices such as mobile phones, laptops, smartphones or other devices that are capable of performing digital data processing and/or video display.

**[0046]** FIG. 2 is a block diagram of a video processing apparatus 3600. The apparatus 3600 may be used to implement one or more of the methods described herein. The apparatus 3600 may be embodied in a smartphone, tablet, computer, Internet of Things (IoT) receiver, and so on. The apparatus 3600 may include one or more processors 3602, one or more memories 3604 and video processing hardware 3606. The processor(s) 3602 may be configured to implement one or more methods described in the present document. The memory (memories) 3604 may be used for storing data and code used for implementing the methods and techniques described herein. The video processing hardware 3606 may be used to implement, in hardware circuitry, some techniques described in the present document.

**[0047]** FIG. 4 is a block diagram that illustrates an example video coding system 100 that may utilize the techniques of this disclosure.

**[0048]** As shown in FIG. 4, video coding system 100 may include a source device 110 and a destination device 120. Source device 110 generates encoded video data which may be referred to as a video encoding device. Destination device 120 may decode the encoded video data generated by source device 110 which may be referred to as a video decoding device.

**[0049]** Source device 110 may include a video source 112, a video encoder 114, and an input/output (I/O) interface 116.

**[0050]** Video source 112 may include a source such as a video capture device, an interface to receive video data from a video content provider, and/or a computer graphics system for generating video data, or a combination of such sources. The video data may comprise one or more pictures. Video encoder 114 encodes the video data from video source 112 to generate a bitstream. The bitstream may include a sequence of bits that form a coded representation of the video data. The bitstream may include coded pictures and associated data. The coded picture is a coded representation of a picture. The associated data may include sequence parameter sets, picture parameter sets, and other syntax structures. I/O interface 116 may include a

modulator/demodulator (modem) and/or a transmitter. The encoded video data may be transmitted directly to destination device 120 via I/O interface 116 through network 130a. The encoded video data may also be stored onto a storage medium/server 130b for access by destination device 120.

**[0051]** Destination device 120 may include an I/O interface 126, a video decoder 124, and a display device 122.

**[0052]** I/O interface 126 may include a receiver and/or a modem. I/O interface 126 may acquire encoded video data from the source device 110 or the storage medium/ server 130b. Video decoder 124 may decode the encoded video data. Display device 122 may display the decoded video data to a user. Display device 122 may be integrated with the destination device 120, or may be external to destination device 120 which be configured to interface with an external display device.

**[0053]** Video encoder 114 and video decoder 124 may operate according to a video compression standard, such as the High Efficiency Video Coding (HEVC) standard, Versatile Video Coding (VVC) standard and other current and/or further standards.

**[0054]** FIG. 5 is a block diagram illustrating an example of video encoder 200, which may be video encoder 114 in the system 100 illustrated in FIG. 4.

**[0055]** Video encoder 200 may be configured to perform any or all of the techniques of this disclosure. In the example of FIG. 5, video encoder 200 includes a plurality of functional components. The techniques described in this disclosure may be shared among the various components of video encoder 200. In some examples, a processor may be configured to perform any or all of the techniques described in this disclosure.

**[0056]** The functional components of video encoder 200 may include a partition unit 201, a predication unit 202 which may include a mode select unit 203, a motion estimation unit 204, a motion compensation unit 205 and an intra prediction unit 206, a residual generation unit 207, a transform unit 208, a quantization unit 209, an inverse quantization unit 210, an inverse transform unit 211, a reconstruction unit 212, a buffer 213, and an entropy encoding unit 214.

**[0057]** In other examples, video encoder 200 may include more, fewer, or different functional components. In an example, predication unit 202 may include an intra block copy (IBC) unit. The IBC unit may perform predication in an IBC mode in which at least one reference picture is a picture where the current video block is located.

**[0058]** Furthermore, some components, such as motion estimation unit 204 and motion compensation unit 205 may be highly integrated, but are represented in the example of FIG. 5 separately for purposes of explanation.

**[0059]** Partition unit 201 may partition a picture into one or more video blocks. Video encoder 200 and video decoder 300 may support various video block sizes.

**[0060]** Mode select unit 203 may select one of the coding modes, intra or inter, e.g., based on error results, and provide the resulting intra- or inter-coded block to a residual generation unit 207 to generate residual block data and to a reconstruction unit 212 to reconstruct the encoded block for use as a reference picture. In some example, Mode select unit 203 may select a combination of intra and inter predication (CIIP) mode in which the predication is based on an inter predication signal and an intra predication signal. Mode select unit 203 may also select a resolution for a motion vector (e.g., a sub-pixel or integer pixel precision) for the block in the case of inter-predication.

**[0061]** To perform inter prediction on a current video block, motion estimation unit 204 may generate motion information for the current video block by comparing one or more reference frames from buffer 213 to the current video block. Motion compensation unit 205 may determine a predicted video block for the current video block based on the motion information and decoded samples of pictures from buffer 213 other than the picture associated with the current video block.

**[0062]** Motion estimation unit 204 and motion compensation unit 205 may perform different operations for a current video block, for example, depending on whether the current video block is in an I slice, a P slice, or a B slice.

**[0063]** In some examples, motion estimation unit 204 may perform uni-directional prediction for the current video block, and motion estimation unit 204 may search reference pictures of list 0 or list 1 for a reference video block for the current video block. Motion estimation unit 204 may then generate a reference index that indicates the reference picture in list 0 or list 1 that contains the reference video block and a motion vector that indicates a spatial displacement between the current video block and the reference video block. Motion estimation unit 204 may output the reference index, a prediction direction indicator, and the motion vector as the motion information of the current video block. Motion compensation unit 205 may generate the predicted video block of the current block based on the reference video block indicated by the motion information of the current video block.

**[0064]** In other examples, motion estimation unit 204 may perform bi-directional prediction for the current video block, motion estimation unit 204 may search the reference pictures in list 0 for a reference video block for the current video block and may also search the reference pictures in list 1 for another reference video block for the current video block. Motion estimation unit 204 may then generate reference indexes that indicate the reference pictures in list 0 and list 1 containing the reference video blocks and motion vectors that indicate spatial displacements between the reference video blocks and the current video block. Motion estimation unit 204 may output the reference indexes and the motion vectors of the current video block as the motion information of the current video block. Motion compensation unit 205 may generate the predicted video block of the current video block based on the reference video blocks indicated by the motion information of the current video block.

**[0065]** In some examples, motion estimation unit 204 may output a full set of motion information for decoding processing of a decoder.

**[0066]** In some examples, motion estimation unit 204 may do not output a full set of motion information for the current video. Rather, motion estimation unit 204 may signal the motion information of the current video block with reference to the motion information of another video block. For example, motion estimation unit 204 may determine that the motion information of the current video block is sufficiently similar to the motion information of a neighboring video block.

**[0067]** In one example, motion estimation unit 204 may indicate, in a syntax structure associated with the current video block, a value that indicates to the video decoder 300 that the current video block has the same motion information as the another video block.

**[0068]** In another example, motion estimation unit 204 may identify, in a syntax structure associated with the current video block, another video block and a motion vector difference (MVD). The motion vector difference indicates a difference between the motion vector of the current video block and the motion vector of the indicated video block. The video decoder 300 may use the motion vector of the indicated video block and the motion vector difference to determine the motion vector of the current video block.

**[0069]** As discussed above, video encoder 200 may predictively signal the motion vector. Two examples of predictive signaling techniques that may be implemented by video encoder 200 include advanced motion vector predication (AMVP) and merge mode signaling.

**[0070]** Intra prediction unit 206 may perform intra prediction on the current video block. When intra prediction unit 206 performs intra prediction on the current video block, intra prediction unit 206 may generate prediction data for the current video block based on decoded samples of other video blocks in the same picture. The prediction data for the current video block may include a predicted video block and various syntax elements.

**[0071]** Residual generation unit 207 may generate residual data for the current video block by subtracting (e.g., indicated by the minus sign) the predicted video block(s) of the current video block from the current video block. The residual data of the current video block may include residual video blocks that correspond to different sample components of the samples in the current video block.

**[0072]** In other examples, there may be no residual data for the current video block for the current video block, for example in a skip mode, and residual generation unit 207 may not perform the subtracting operation.

**[0073]** Transform processing unit 208 may generate one or more transform coefficient video blocks for the current video block by applying one or more transforms to a residual video block associated with the current video block.

**[0074]** After transform processing unit 208 generates a transform coefficient video block associated with the current video block, quantization unit 209 may quantize the transform coefficient video block associated with the current video block based on one or more quantization parameter (QP) values associated with the current video block.

**[0075]** Inverse quantization unit 210 and inverse transform unit 211 may apply inverse quantization and inverse transforms to the transform coefficient video block, respectively, to reconstruct a residual video block from the transform coefficient video block. Reconstruction unit 212 may add the reconstructed residual video block to corresponding samples from one or more predicted video blocks generated by the predication unit 202 to produce a reconstructed video block associated with the current block for storage in the buffer 213.

**[0076]** After reconstruction unit 212 reconstructs the video block, loop filtering operation may be performed reduce video blocking artifacts in the video block.

**[0077]** Entropy encoding unit 214 may receive data from other functional components of the video encoder 200. When entropy encoding unit 214 receives the data, entropy encoding unit 214

may perform one or more entropy encoding operations to generate entropy encoded data and output a bitstream that includes the entropy encoded data.

**[0078]** FIG. 6 is a block diagram illustrating an example of video decoder 300 which may be video decoder 114 in the system 100 illustrated in FIG. 4.

**[0079]** The video decoder 300 may be configured to perform any or all of the techniques of this disclosure. In the example of FIG. 5, the video decoder 300 includes a plurality of functional components. The techniques described in this disclosure may be shared among the various components of the video decoder 300. In some examples, a processor may be configured to perform any or all of the techniques described in this disclosure.

**[0080]** In the example of FIG. 6, video decoder 300 includes an entropy decoding unit 301, a motion compensation unit 302, an intra prediction unit 303, an inverse quantization unit 304,an inverse transformation unit 305 , and a reconstruction unit 306 and a buffer 307. Video decoder 300 may, in some examples, perform a decoding pass generally reciprocal to the encoding pass described with respect to video encoder 200 (FIG. 5).

**[0081]** Entropy decoding unit 301 may retrieve an encoded bitstream. The encoded bitstream may include entropy coded video data (e.g., encoded blocks of video data). Entropy decoding unit 301 may decode the entropy coded video data, and from the entropy decoded video data, motion compensation unit 302 may determine motion information including motion vectors, motion vector precision, reference picture list indexes, and other motion information. Motion compensation unit 302 may, for example, determine such information by performing the AMVP and merge mode.

**[0082]** Motion compensation unit 302 may produce motion compensated blocks, possibly performing interpolation based on interpolation filters. Identifiers for interpolation filters to be used with sub-pixel precision may be included in the syntax elements.

**[0083]** Motion compensation unit 302 may use interpolation filters as used by video encoder 20 during encoding of the video block to calculate interpolated values for sub-integer pixels of a reference block. Motion compensation unit 302 may determine the interpolation filters used by video encoder 200 according to received syntax information and use the interpolation filters to produce predictive blocks.

**[0084]** Motion compensation unit 302 may uses some of the syntax information to determine sizes of blocks used to encode frame(s) and/or slice(s) of the encoded video sequence, partition

information that describes how each macroblock of a picture of the encoded video sequence is partitioned, modes indicating how each partition is encoded, one or more reference frames (and reference frame lists) for each inter-encoded block, and other information to decode the encoded video sequence.

**[0085]** Intra prediction unit 303 may use intra prediction modes for example received in the bitstream to form a prediction block from spatially adjacent blocks. Inverse quantization unit 303 inverse quantizes, i.e., de-quantizes, the quantized video block coefficients provided in the bitstream and decoded by entropy decoding unit 301. Inverse transform unit 303 applies an inverse transform.

**[0086]** Reconstruction unit 306 may sum the residual blocks with the corresponding prediction blocks generated by motion compensation unit 202 or intra-prediction unit 303 to form decoded blocks. If desired, a deblocking filter may also be applied to filter the decoded blocks in order to remove blockiness artifacts. The decoded video blocks are then stored in buffer 307, which provides reference blocks for subsequent motion compensation/intra predication and also produces decoded video for presentation on a display device.

**[0087]** A listing of solutions describes some embodiments of the disclosed technology.

**[0088]** A first set of solutions is provided next. The following solutions show example embodiments of techniques discussed in the previous section (e.g., items 1-9).

**[0089]** 1. A method of video processing (e.g., method 600 in FIG. 3), comprising performing (602) a conversion between a video comprising one or more video layers comprising one or more video pictures and a coded representation of the video, wherein the coded representation conforms to a format rule related to extraction of a sub-bitstream from the coded representation.

**[0090]** 2. The method of solution 1, further comprising: extracting the sub-bitstream from the coded representation according to the format rule.

**[0091]** The following solutions show example embodiments of techniques discussed in the previous section (e.g., item 1)

**[0092]** 3. The method of any of solutions 1-2, wherein, during extracting the sub-bitstream, a target id used for the extracting is allowed to be between range 0 to a value of a syntax field indicating in a video parameter set for the coded representation.

**[0093]** The following solutions show example embodiments of techniques discussed in the previous section (e.g., item 2)

**[0094]** 4. The method of any of solutions 1-3, wherein the sub-bitstream is extracted without removing an access unit delimiter network abstraction layer (AUD NAL) from an output bitstream according to a layer id.

**[0095]** The following solutions show example embodiments of techniques discussed in the previous section (e.g., item 3)

**[0096]** 5. The method of any of solutions 1-4, wherein the sub-bitstream is extracted by selectively removing network abstraction layer units that include a scalable nesting supplementary enhancement information message that are not applicable to output layers being extracted.

**[0097]** The following solutions show example embodiments of techniques discussed in the previous section (e.g., item 5)

**[0098]** 6. The method of any of solutions 1-5, wherein the sub-bitstream is extracted by constraining the extracting to generate a non-scalable nested supplemental enhancement information (SEI) from a scalable nesting SEI message using a flag for output layer set being set and a flag for subpicture being disabled.

**[0099]** The following solutions show example embodiments of techniques discussed in the previous section (e.g., item 6)

**[00100]** 7. The method of any of solutions 1-6, wherein the sub-bitstream is extracted according to a rule specifying extraction of multiple scalable-nested supplemental enhancement information (SEI) messages from a single SEI network abstraction layer unit.

**[00101]** The following solutions show example embodiments of techniques discussed in the previous section (e.g., item 7)

**[00102]** 8. The method of any of solutions 1-7, wherein the sub-bitstream is extracted according to a rule that removes, from the coded representation, all supplemental enhancement information network abstraction layer SEI NAL units from which some SEI messages have been extracted and included as non-scalable-nested SEI messages.

**[00103]** The following solutions show example embodiments of techniques discussed in the previous section (e.g., item 8)

**[00104]** 9. The method of any of solutions 1-8, wherein the format rule specifies that when a supplemental enhancement information network abstraction layer (SEI NAL) unit contain an SEI message with payloadType equal to 0, 1, or 130, the SEI NAL unit is not allowed to contain an SEI message with payloadType not equal to 0 (BP), 1 (PT), 130 (DUI), or 133 (scalable nesting).

**[00105]** 10. The method of any of solutions 1-9, wherein the performing the conversion comprises encoding the video into the coded representation.

**[00106]** 11. The method of any of solutions 1-9, wherein the performing the conversion comprises parsing and decoding the coded representation to generate the video.

**[00107]** 12. A video decoding apparatus comprising a processor configured to implement a method recited in one or more of solutions 1 to 11.

**[00108]** 13. A video encoding apparatus comprising a processor configured to implement a method recited in one or more of solutions 1 to 11.

**[00109]** 14. A computer program product having computer code stored thereon, the code, when executed by a processor, causes the processor to implement a method recited in any of solutions 1 to 11.

**[00110]** 15. A method, apparatus or system described in the present document.

**[00111]** A second set of solutions show example embodiments of techniques discussed in the previous section (e.g., items 1, 4, 8, and 9).

**[00112]** 1. A method of video processing (e.g., method 700 as shown in FIG. 7A), comprising: performing 702 a conversion between a video comprising multiple layers and a bitstream of the video according to a rule, wherein the rule specifies a maximum allowed value of a temporal layer identifier value of a sub-bitstream that is extracted from the bitstream.

**[00113]** 2. The method of solution 1, wherein the rule is responsive to a number of layers in the bitstream.

**[00114]** 3. The method of solution 1 or 2, wherein, in case that the number of layers is greater than 1, the rule specifies that the maximum allowed value of the temporal layer identifier is in a range of zero to a value based on a first syntax element in a video parameter set referenced by the bitstream.

**[00115]** 4. The method of solution 1 or 2, wherein, in case that the number of layers is equal to 1, the rule specifies that the maximum allowed value of the temporal layer identifier is in a range of zero to a value based on a second syntax element in a sequence parameter set referenced by the bitstream.

**[00116]** 5. The method of solution 3, wherein the first syntax element specifies a maximum number of temporal sublayers that is allowed to be present in a layer specified by the video parameter set minus 1.

**[00117]** 6. The method of solution 4, wherein the second syntax element specifies a maximum number of temporal sublayers that is allowed to be present in a layer specified by the sequence parameter set minus 1.

**[00118]** 7. The method of solution 3, wherein the first syntax element in the video parameter set is vps_max_sublayers_minus1.

**[00119]** 8. The method of solution 4, wherein the second syntax element in the sequence parameter set is sps_max_sublayers_minus1.

**[00120]** 9. The method of solution 3 or 4, wherein the value is equal to a value of the first syntax element or the second syntax element.

**[00121]** 10. A method of video processing (e.g., method 710 as shown in FIG. 7B), comprising: performing 712 a conversion between a video and a bitstream of the video according to a rule, wherein the rule defines network abstraction layer (NAL) units to be extracted from the bitstream during a sub-bitstream extraction process to output a sub-bitstream, and wherein the rule specifies to derive the sub-bitstream based on whether a list of NAL unit header identifier values in an output layer set (OLS) with a target OLS index does not include all values of NAL unit header identifiers in all video coding layer (VCL) NAL units in the bitstream that is input to the sub-bitstream extraction process.

**[00122]** 11. The method of solution 10, wherein the rule specifies to remove from the sub-bitstream all supplemental enhancement information (SEI) NAL units that contain a non-scalable-nested SEI message with payload type equal to 0 or 130, in case that the list of NAL unit header identifier values in the output layer set does not include all values of NAL unit header identifiers in all VCL NAL units in the bitstream.

**[00123]** 12. A method of video processing (e.g., method 720 as shown in FIG. 7C), comprising: performing 722 a conversion between a video and a bitstream of the video according to a rule, wherein the rule defines network abstraction layer (NAL) units to be extracted from the bitstream during a sub-bitstream extraction process to output a sub-bitstream, and wherein the rule specifies, responsive to a payload type of a first supplemental enhancement information (SEI) message included in an SEI network abstraction layer (NAL) unit, to disallow the SEI NAL unit to contain an SEI message with a certain payload type.

**[00124]** 13. The method of solution 12, wherein the payload type of the first SEI message is 0, 1, or 130 and the rule specifies to disallow the SEI NAL unit to contain the SEI message with the certain payload type that is not equal to 0, 1, 130, or 133.

**[00125]** 14. A method of video processing (e.g., method 730 as shown in FIG. 7D), comprising: performing 732 a conversion between a video and a bitstream of the video according to a format rule, wherein the format rule specifies that the bitstream includes a flag specifying whether one or more non-scalable nested supplemental enhancement information (SEI) messages with one or more particular payload types apply to all output layer sets referenced by a coding layer.

**[00126]** 15. The method of solution 14, wherein the one or more particular payload types are equal to 1 and 130.

**[00127]** 16. The method of solution 14, wherein the one or more non-scalable SEI messages are picture timing (PT) SEI messages and decoding unit information (DUI) SEI messages.

**[00128]** 17. The method of solution 14, wherein the one or more particular payload types are equal to 0, 1 and 130.

**[00129]** 18. The method of solution 14, wherein the one or more non-scalable SEI messages are buffering period (BP) SEI messages and decoding unit information (DUI) SEI messages.

**[00130]** 19. The method of solution 14, wherein the one or more particular payload type is equal to 130.

**[00131]** 20. The method of solution 14, wherein the one or more non-scalable SEI messages are decoding unit information (DUI) SEI messages.

**[00132]** 21. The method of solution 14, wherein the one or more particular payload type is equal to 0.

**[00133]** 22. The method of solution 1, wherein the one or more non-scalable SEI messages are buffering period (BP) SEI messages.

**[00134]** 23. The method of any of solutions 1 to 22, wherein the conversion includes encoding the video into the bitstream.

**[00135]** 24. The method of any of solutions 1 to 22, wherein the conversion includes decoding the video from the bitstream.

**[00136]** 25. The method of any of solutions 1 to 22, wherein the conversion includes generating the bitstream from the video, and the method further comprises: storing the bitstream in a non-transitory computer-readable recording medium.

**[00137]** 26. A video processing apparatus comprising a processor configured to implement a method recited in any one or more of solutions 1 to 25.

**[00138]** 27. A method of storing a bitstream of a video, comprising, a method recited in any one of solutions 1 to 25, and further including storing the bitstream to a non-transitory computer-readable recording medium.

**[00139]** 28. A computer readable medium storing program code that, when executed, causes a processor to implement a method recited in any one or more of solutions 1 to 25.

**[00140]** 29. A computer readable medium that stores a bitstream generated according to any of the above described methods.

**[00141]** 30. A video processing apparatus for storing a bitstream representation, wherein the video processing apparatus is configured to implement a method recited in any one or more of solutions 1 to 25.

**[00142]** A third set of solutions show example embodiments of techniques discussed in the previous section (e.g., item 2).

**[00143]** 1. A method of video processing (e.g., method 800 as shown in FIG. 8), comprising: performing 802 a conversion between a video and a bitstream of the video according to a rule, wherein the rule specifies a sub-bitstream extraction process by which an output sub-bitstream is extracted from the bitstream without removing network abstraction layer (NAL) units of a particular type and having a particular NAL unit header identifier value, wherein the particular type includes an access unit delimiter (AUD) NAL unit.

**[00144]** 2. The method of solution 1, wherein the particular type includes a video parameter set (VPS) NAL unit.

**[00145]** 3. The method of solution 1 or 2, wherein the particular type includes a decoding capability information NAL unit.

**[00146]** 4. The method of any of solutions 1-3, wherein the particular type includes an end of bitstream NAL unit.

**[00147]** 5. The method of any of solutions 1-4, wherein the particular type includes a supplemental enhancement information NAL unit containing a non-scalable nested SEI massage with a payload type that is equal to 0, 1, 130 or 203.

**[00148]** 6. The method of any of solutions 1-6, wherein the particular NAL unit header identifier value includes a layer identifier value that is included in a list of layer values for the output sub-bitstream.

**[00149]** 7. The method of solution 1, wherein the rule specifies that a particular type NAL unit is not removed from the output bitstream according to the exception regardless of a NAL unit header identifier value associated with the particular type NAL unit.

**[00150]** 8. The method of any of solutions 1 to 7, wherein the conversion includes encoding the video into the bitstream.

**[00151]** 9. The method of any of solutions 1 to 7, wherein the conversion includes decoding the video from the bitstream.

**[00152]** 10. The method of any of solutions 1 to 7, wherein the conversion includes generating the bitstream from the video, and the method further comprises: storing the bitstream in a non-transitory computer-readable recording medium.

**[00153]** 11. A video processing apparatus comprising a processor configured to implement a method recited in any one or more of solutions 1 to 10.

**[00154]** 12. A method of storing a bitstream of a video, comprising, a method recited in any one of solutions 1 to 10, and further including storing the bitstream to a non-transitory computer-readable recording medium.

**[00155]** 13. A computer readable medium storing program code that, when executed, causes a processor to implement a method recited in any one or more of solutions 1 to 10.

**[00156]** 14. A computer readable medium that stores a bitstream generated according to any of the above described methods.

**[00157]** 15. A video processing apparatus for storing a bitstream representation, wherein the video processing apparatus is configured to implement a method recited in any one or more of solutions 1 to 10.

**[00158]** A fourth set of solutions show example embodiments of techniques discussed in the previous section (e.g., item 3).

**[00159]** 1. A method of video processing (e.g., method 900 as shown in FIG. 9), comprising: performing 902 a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies, in a process of sub-bitstream extraction, to remove network abstraction layer (NAL) units that include a scalable nesting supplementary

enhancement information (SEI) message applied to layers that are not included in a target output layer set (OLS).

**[00160]** 2. The method of solution 1, wherein the scalable nesting SEI message is associated with a flag having a value equal to a particular value that specifies the scalable nesting SEI message applies to the layers.

**[00161]** 3. The method of solution 2, wherein the particular value is 0.

**[00162]** 4. The method of solution 1, wherein the rule further specifies to remove the NAL units that include the SEI message having no value in a first list (NestingLayerID[i]) that is in a second list (LayerIdInOls[ targetOlsIdx ]), whereby the first list specifies NAL unit header identifier values of layers to which the scalable nesting SEI message applies and i is in a range of 0 to NumLayers − 1, NumLayers indicating a number of the layers to which the scalable nesting SEI message applies, and the second list specifies the NAL unit header identifier values in the target output layer set with a target output layer index.

**[00163]** 5. The method of any of solutions 1 to 4, wherein the conversion includes encoding the video into the bitstream.

**[00164]** 6. The method of any of solutions 1 to 4, wherein the conversion includes decoding the video from the bitstream.

**[00165]** 7. The method of any of solutions 1 to 4, wherein the conversion includes generating the bitstream from the video, and the method further comprises: storing the bitstream in a non-transitory computer-readable recording medium.

**[00166]** 8. A video processing apparatus comprising a processor configured to implement a method recited in any one or more of solutions 1 to 7.

**[00167]** 9. A method of storing a bitstream of a video, comprising, a method recited in any one of solutions 1 to 7, and further including storing the bitstream to a non-transitory computer-readable recording medium.

**[00168]** 10. A computer readable medium storing program code that, when executed, causes a processor to implement a method recited in any one or more of solutions 1 to 7.

**[00169]** 11. A computer readable medium that stores a bitstream generated according to any of the above described methods.

**[00170]** 12. A video processing apparatus for storing a bitstream representation, wherein the video processing apparatus is configured to implement a method recited in any one or more of

solutions 1 to 7.

**[00171]** A fifth set of solutions show example embodiments of techniques discussed in the previous section (e.g., items 5-7).

**[00172]** 1. A method of video processing (e.g., method 1000 as shown in FIG. 10A), comprising: performing 1002 a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies, in a process of sub-bitstream extraction, a non-scalable nested supplementary enhancement information (SEI) message is generated by extracting an SEI message that is scalably-nested from a scalable nesting SEI message based on a first flag indicating whether the SEI message applies to specific output layer sets (OLSs) and a second flag indicating whether the SEI message applies to all subpictures or only to specific subpictures.

**[00173]** 2. The method of solution 1, wherein the rule specifies to generate the non-scalable nested SEI message responsive to conditions i) the first flag having a first value that specifies that the SEI message applies to specific output layer sets (OLSs) and ii) the second flag having a second value that specifies that the SEI message that applies to the specific OLSs applies to all subpictures of the specified OLSs being satisfied.

**[00174]** 3. A method of video processing (e.g., method 1010 as shown in FIG. 10B), comprising: performing 1012 a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies, in a process of sub-bitstream extraction, a non-scalable nested supplemental enhancement information (SEI) message is generated by extracting multiple scalable-nested SEI messages from a first SEI network abstraction layer (NAL) unit in a picture unit.

**[00175]** 4. The method of solution 3, wherein the multiple scalable-nested SEI messages are included in a second SEI NAL unit included in the picture unit.

**[00176]** 5. The method of solution 4, wherein the second SEI NAL unit is immediately after the first SEI NAL unit.

**[00177]** 6. A method of video processing (e.g., method 1020 as shown in FIG. 10C), comprising: performing 1022 a conversion between a video including one or more layers and a bitstream of the video according to a rule, wherein the rule specifies a sub-bitstream extraction process to generate an output bitstream, wherein the rule specifies handling of one or more supplemental

enhancement information (SEI) network abstraction layer (NAL) units during the sub-bitstream extraction process.

**[00178]** 7. The method of solution 6, wherein the rule specifies to remove the one or more SEI NAL units from which some SEI messages have been extracted and included as non-scalable nested SEI messages.

**[00179]** 8. The method of solution 6, wherein the rule specifies to remove a SEI NAL unit that contains a scalable-nested SEI message applying only to a target output layer set.

**[00180]** 9. The method of solution 6, wherein the rule specifies to remove a SEI NAL unit that contains a scalable nested SEI message that applies to output layer sets (OLSs) in which there is no OLS other than a target OLS.

**[00181]** 10. The method of any of solutions 1 to 9, wherein the conversion includes encoding the video into the bitstream.

**[00182]** 11. The method of any of solutions 1 to 9, wherein the conversion includes decoding the video from the bitstream.

**[00183]** 12. The method of any of solutions 1 to 9, wherein the conversion includes generating the bitstream from the video, and the method further comprises: storing the bitstream in a non-transitory computer-readable recording medium.

**[00184]** 13. A video processing apparatus comprising a processor configured to implement a method recited in any one or more of solutions 1 to 12.

**[00185]** 14. A method of storing a bitstream of a video, comprising, a method recited in any one of solutions 1 to 12, and further including storing the bitstream to a non-transitory computer-readable recording medium.

**[00186]** 15. A computer readable medium storing program code that, when executed, causes a processor to implement a method recited in any one or more of solutions 1 to 12.

**[00187]** 16. A computer readable medium that stores a bitstream generated according to any of the above described methods.

**[00188]** 17. A video processing apparatus for storing a bitstream representation, wherein the video processing apparatus is configured to implement a method recited in any one or more of solutions 1 to 12.

**[00189]** The disclosed and other solutions, examples, embodiments, modules and the functional operations described in this document can be implemented in digital electronic circuitry, or in

computer software, firmware, or hardware, including the structures disclosed in this document and their structural equivalents, or in combinations of one or more of them. The disclosed and other embodiments can be implemented as one or more computer program products, i.e., one or more modules of computer program instructions encoded on a computer readable medium for execution by, or to control the operation of, data processing apparatus. The computer readable medium can be a machine-readable storage device, a machine-readable storage substrate, a memory device, a composition of matter effecting a machine-readable propagated signal, or a combination of one or more them. The term "data processing apparatus" encompasses all apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them. A propagated signal is an artificially generated signal, e.g., a machine-generated electrical, optical, or electromagnetic signal, that is generated to encode information for transmission to suitable receiver apparatus.

**[00190]** A computer program (also known as a program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, and it can be deployed in any form, including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A computer program does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

**[00191]** The processes and logic flows described in this document can be performed by one or more programmable processors executing one or more computer programs to perform functions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit).

**[00192]** Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read only memory or a random-access memory or both. The essential elements of a computer are a processor for performing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Computer readable media suitable for storing computer program instructions and data include all forms of non-volatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto optical disks; and CD ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

**[00193]** While this patent document contains many specifics, these should not be construed as limitations on the scope of any subject matter or of what may be claimed, but rather as descriptions of features that may be specific to particular embodiments of particular techniques. Certain features that are described in this patent document in the context of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

**[00194]** Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. Moreover, the separation of various system components in the embodiments described in this patent document should not be understood as requiring such separation in all embodiments.

**[00195]** Only a few implementations and examples are described and other implementations, enhancements and variations can be made based on what is described and illustrated in this patent document.

# CLAIMS

1. A method of processing video data, comprising:

performing a conversion between a video including one or more layers and a bitstream of the video according to a rule,

wherein the rule specifies, in a process of sub-bitstream extraction in which a first sub-bitstream is used to derive an output sub-bitstream, whether to generate non-scalable-nested supplementary enhancement information (SEI) messages by extracting scalable-nested SEI messages from a scalable nesting SEI message is based on a first flag denoted as sn_ols_flag and a second flag denoted as sn_subpic_flag, wherein the scalable nesting SEI message is contained in a first SEI network abstraction layer (NAL) unit contained in the first sub-bitstream, and

wherein in a case that the non-scalable-nested SEI messages are generated, a second SEI NAL unit is generated, and the scalable-nested SEI messages are extracted from the scalable nesting SEI message and are included in the second SEI NAL unit as the non-scalable-nested SEI messages.

2. The method of claim 1, wherein the second SEI NAL unit is included immediately after the first SEI NAL unit in a PU containing the first SEI NAL unit in the first sub-bitstream.

3. The method of claim 2, wherein the first SEI NAL unit is removed from the first sub-bitstream to generate the output sub-bitstream.

4. The method of claim 1, wherein the rule specifies to generate the non-scalable-nested SEI messages when a set of conditions are satisfied, wherein the set of conditions comprise that the first SEI NAL unit contain the scalable nesting SEI message with the sn_ols_flag equal to 1 and the sn_subpic_flag that applies to a target OLS is equal to 0.

5. The method of claim 4, wherein the set of conditions comprise that a list of network abstraction layer (NAL) unit header identifier values in an OLS with a target OLS index does not include all

values of NAL unit header identifiers in all video coding layer (VCL) NAL units in the first sub-bitstream.

6. The method of claim 5, wherein the rule further specifies that, in response to that the list of NAL unit header identifier values in the OLS does not include all values of NAL unit header identifiers in all VCL NAL units in the bitstream that is input to the process of sub-bitstream extraction, the following further applies:

1) all SEI NAL units that contain a non-scalable-nested SEI message with payload type equal to 0 or 130 are removed from the output sub-bitstream, and

2) when general_same_pic_timing_in_all_ols_flag is equal to 0, all SEI NAL units that contain a non-scalable-nested SEI message with payload type equal to 1 are removed from the output sub-bitstream.

7. The method of claim 1, wherein the conversion includes encoding the video into the bitstream.

8. The method of claim 1, wherein the conversion includes decoding the video from the bitstream.

9. An apparatus for processing video data comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to:

perform a conversion between a video including one or more layers and a bitstream of the video according to a rule,

wherein the rule specifies, in a process of sub-bitstream extraction in which a first sub-bitstream is used to derive an output sub-bitstream, whether to generate non-scalable-nested supplementary enhancement information (SEI) messages by extracting scalable-nested SEI messages from a scalable nesting SEI message is based on a first flag denoted as sn_ols_flag and a second flag denoted as sn_subpic_flag,

wherein the scalable nesting SEI message is contained in a first SEI network abstraction layer (NAL) unit contained in the first sub-bitstream, and

wherein in a case that the non-scalable-nested SEI messages are generated, a second SEI NAL unit is generated, and the scalable-nested SEI messages are extracted from the scalable

nesting SEI message and are included in the second SEI NAL unit as the non-scalable-nested SEI messages.

10. The apparatus of claim 9,

wherein the second SEI NAL unit is included immediately after the first SEI NAL unit in a PU containing the first SEI NAL unit in the first sub-bitstream, and

wherein the first SEI NAL unit is removed from the first sub-bitstream to generate the output sub-bitstream.

11. The apparatus of claim 9, wherein the rule specifies to generate the non-scalable-nested SEI messages when a set of conditions are satisfied, wherein the set of conditions comprise that the first SEI NAL unit contain the scalable nesting SEI message with the sn_ols_flag equal to 1 and the sn_subpic_flag that applies to a target OLS is equal to 0.

12. The apparatus of claim 11, wherein the set of conditions comprise that a list of network abstraction layer (NAL) unit header identifier values in an OLS with a target OLS index does not include all values of NAL unit header identifiers in all video coding layer (VCL) NAL units in the first sub-bitstream.

13. The apparatus of claim 12, wherein the rule further specifies that, in response to that the list of NAL unit header identifier values in the OLS does not include all values of NAL unit header identifiers in all VCL NAL units in the bitstream that is input to the process of sub-bitstream extraction, the following further applies:

1) all SEI NAL units that contain a non-scalable-nested SEI message with payload type equal to 0 or 130 are removed from the output sub-bitstream, and

2) when general_same_pic_timing_in_all_ols_flag is equal to 0, all SEI NAL units that contain a non-scalable-nested SEI message with payload type equal to 1 are removed from the output sub-bitstream.

14. A non-transitory computer-readable storage medium storing instructions that cause a processor to:

perform a conversion between a video including one or more layers and a bitstream of the video according to a rule,

wherein the rule specifies, in a process of sub-bitstream extraction in which a first sub-bitstream is used to derive an output sub-bitstream, whether to generate non-scalable-nested supplementary enhancement information (SEI) messages by extracting scalable-nested SEI messages from a scalable nesting SEI message is based on a first flag denoted as sn_ols_flag and a second flag denoted as sn_subpic_flag,

wherein the scalable nesting SEI message is contained in a first SEI network abstraction layer (NAL) unit contained in the first sub-bitstream, and

wherein in a case that the non-scalable-nested SEI messages are generated, a second SEI NAL unit is generated, and the scalable-nested SEI messages are extracted from the scalable nesting SEI message and are included in the second SEI NAL unit as the non-scalable-nested SEI messages.

15. The non-transitory computer-readable storage medium of claim 14,

wherein the second SEI NAL unit is included immediately after the first SEI NAL unit in a PU containing the first SEI NAL unit in the first sub-bitstream, and

wherein the first SEI NAL unit is removed from the first sub-bitstream to generate the output sub-bitstream.

16. A non-transitory computer-readable recording medium storing a bitstream of a video which is generated by a method performed by a video processing apparatus, wherein the method comprises:

generating the bitstream of the video including one or more layers according to a rule,

wherein the rule specifies, in a process of sub-bitstream extraction in which a first sub-bitstream is used to derive an output sub-bitstream, whether to generate non-scalable-nested supplementary enhancement information (SEI) messages by extracting scalable-nested SEI messages from a scalable nesting SEI message is based on a first flag denoted as sn_ols_flag and a second flag denoted as sn_subpic_flag,

wherein the scalable nesting SEI message is contained in a first SEI network abstraction layer (NAL) unit contained in the first sub-bitstream, and

wherein in a case that the non-scalable-nested SEI messages are generated, a second SEI NAL unit is generated, and the scalable-nested SEI messages are extracted from the scalable nesting SEI message and are included in the second SEI NAL unit as the non-scalable-nested SEI messages.

17. The non-transitory computer-readable recording medium of claim 16,

wherein the second SEI NAL unit is included immediately after the first SEI NAL unit in a PU containing the first SEI NAL unit in the first sub-bitstream, and

wherein the first SEI NAL unit is removed from the first sub-bitstream to generate the output sub-bitstream.