|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  11th Meeting: Shanghai, CN, 10–19 Oct. 2012 | Document: JCTVC-K1007 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **NAL unit header and parameter set designs for HEVC extensions** | | |
| *Status:* | Output Document of JCT-VC | | |
| *Purpose:* | Draft | | |
| *Author(s) or Contact(s):* | Jill Boyce  Ye-Kui Wang | Email: [jill@vidyo.com](mailto:jill@vidyo.com) [yekuiw@qti.qualcomm.com](mailto:yekuiw@qti.qualcomm.com) |  |
| *Source:* | Editor | | |

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

# Abstract

This document includes the specifications of the NAL unit header and parameter set designs for HEVC 3DV (including MV-HEVC) and scalable coding extensions agreed at the 11th JCT-VC meeting.

# NAL unit syntax and semantics

Syntax changes relative to the NAL unit syntax in the HEVC base specification are highlighted.

|  |  |
| --- | --- |
| nal\_unit( NumBytesInNALunit ) { | Descriptor |
| **forbidden\_zero\_bit** | f(1) |
| **nal\_unit\_type** | u(6) |
| **nuh\_layer\_id** // nuh\_reserved\_zero\_6bits in the base spec | u(6) |
| **nuh\_temporal\_id\_plus1** | u(3) |
| NumBytesInRBSP = 0 |  |
| for( i = 2; i < NumBytesInNALunit; i++ ) { |  |
| if( i + 2 < NumBytesInNALunit && next\_bits( 24 ) = = 0x000003 ) { |  |
| **rbsp\_byte[** NumBytesInRBSP++ **]** | b(8) |
| **rbsp\_byte[** NumBytesInRBSP++ **]** | b(8) |
| i += 2 |  |
| **emulation\_prevention\_three\_byte** /\* equal to 0x03 \*/ | f(8) |
| } else |  |
| **rbsp\_byte[** NumBytesInRBSP++ **]** | b(8) |
| } |  |
| } |  |

**nuh\_layer\_id** specifies the identifier of the layer.

Table 7‑1 – NAL unit type codes and NAL unit type classes

|  |  |  |  |
| --- | --- | --- | --- |
| nal\_unit\_type | Name of nal\_unit\_type | Content of NAL unit and RBSP syntax structure | NAL unit type class |
| 15 | BASE\_NUT | Encapsulation of a non-HEVC base layer | VCL |
| 16..20 | RSV\_VCL15.. RSV\_VCL20 | Reserved | VCL |
|  |  |  |  |

**nuh\_temporal\_id\_plus1** minus 1 specifies a temporal identifier for the NAL unit.

The variable TemporalId is specified as

TemporalId = nuh\_temporal\_id\_plus1 − 1 (7‑3)

If nal\_unit\_type is in the range of 7 to 12 (coded slice of a RAP picture), inclusive, TemporalId shall be equal to 0; otherwise, when nal\_unit\_type is in the range of 3 to 6, inclusive (coded slice of a TSA or STSA picture), TemporalId shall not be equal to 0.

The value of TemporalId shall be the same for all VCL NAL units of an access unit with nal\_unit\_type not equal to BASE\_NUT. The value of TemporalId of an access unit is the value of the TemporalId of the VCL NAL units of the access unit.

The value of TemporalId for non-VCL NAL units is constrained as follows:

– If nal\_unit\_type is equal to VPS\_NUT, SPS\_NUT, EOS\_NUT, or EOB\_NUT, TemporalId shall be equal to 0.

– Otherwise, if nal\_unit\_type is equal to AUD\_NUT or FD\_NUT, TemporalId shall be equal to the TemporalId of the access unit containing the non-VCL NAL unit.

– Otherwise, ~~when~~ if nal\_unit\_type is equal to SEI\_NUT, TemporalId shall be greater or equal to the TemporalId of the access unit containing the NAL unit.

– Otherwise, when nal\_unit\_type is equal to BASE\_NUT and avc\_base\_codec\_flag is equal to 1, when temporal\_id is present in the Rec. ITU-T H.264 | ISO/IEC 14496-10 conforming base layer, TemporalId shall be equal to the value of temporal\_id.

Semantics for other syntax elements are as specified in the HEVC base specification.

# Video parameter set

## Video parameter set RBSP syntax and semantics

Syntax changes relative to the video parameter set RBSP syntax in the HEVC base specification are highlighted.

|  |  |
| --- | --- |
| video\_parameter\_set\_rbsp( ) { | Descriptor |
| **video\_parameter\_set\_id** | u(4) |
| **vps\_temporal\_id\_nesting\_flag** | u(1) |
| **vps\_reserved\_zero\_2bits** | u(2) |
| **vps\_max\_num\_layers\_minus1** //reserved\_zero\_6bits in the base spec | u(6) |
| **vps\_max\_sub\_layers\_minus1** | u(3) |
| **next\_essential\_info\_byte\_offset** //vps\_reserved\_zero\_16bits in the base spec | u(16) |
| profile\_tier\_level( 1, vps\_max\_sub\_layers\_minus1 ) |  |
| for( i = 0; i <= vps\_max\_sub\_layers\_minus1; i++ ) { |  |
| **vps\_max\_dec\_pic\_buffering[** i **]** | ue(v) |
| **vps\_max\_num\_reorder\_pics[** i **]** | ue(v) |
| **vps\_max\_latency\_increase[** i **]** | ue(v) |
| } |  |
| **vps\_num\_hrd\_parameters** | ue(v) |
| **vps\_max\_nuh\_reserved\_zero\_layer\_id** | u(6) |
| for( i = 0; i < vps\_num\_hrd\_parameters; i++ ) { |  |
| if( i > 0 ) |  |
| operation\_point\_layer\_ids( i ) |  |
| hrd\_parameters( i = = 0, vps\_max\_sub\_layers\_minus1 ) |  |
| } |  |
| **bit\_equal\_to\_one** | u(1) |
| vps\_extension( ) |  |
| **vps\_extension\_flag** | u(1) |
| if( vps\_extension\_flag ) |  |
| while( more\_rbsp\_data( ) ) |  |
| **vps\_extension\_data\_flag** | u(1) |
| rbsp\_trailing\_bits( ) |  |
| } |  |

**max\_num\_layers\_minus1** plus 1 specifies the maximum number of layers in the coded video sequences referring to the video parameter set.

**next\_essential\_info\_byte\_offset** specifies the byte offset of the next set of fixed-length coded information in the video parameter set NAL unit, starting from the beginning of the NAL unit.

NOTE 3 –Video parameter set information for non-base layer or view starts from a byte-aligned position of the video parameter set NAL unit, with fixed-length coded information that is essential for session negotiation and/or capability exchange. The byte offset specified by next\_essential\_info\_byte\_offset would then help to locate and access those essential information in the video parameter set NAL unit without the need of entropy decoding, which may not be equipped with some network entities that may desire to access only the information in the video parameter set that is essential for session negotiation and/or capability exchange.

**max\_num\_layers\_minus1** plus 1 specifies the maximum number of layers in the coded video sequences referring to the video parameter set.

**bit\_equal\_to\_one** is equal to 1.

Semantics for other syntax elements are as specified in the HEVC base specification.

## Video parameter set extension syntax and semantics

Additions and removals in relative to the vps\_extension( ) syntax structure of Approach 1 in JCTVC-J1007 are highlighted and strikethrough, respectively.

|  |  |
| --- | --- |
| vps\_extension( ) { | Descriptor |
| while( !byte\_aligned( ) ) |  |
| **vps\_extension\_byte\_alignment\_reserved\_one\_bit** | u(1) |
| **avc\_base\_codec\_flag** | u(1) |
| **scalability\_mask** | u(16) |
| for( i = 0; i <NumScalabilityTypes; i++ ) { |  |
| **dimension\_id\_len\_minus1**[ i ] | u(3) |
| } |  |
| **vps\_nuh\_layer\_id\_present\_flag** | u(1) |
| // layer specific information |  |
| for( i = 1; i <= vps\_max\_layers\_minus1; i++ ) { |  |
| // mapping of layer ID to scalability dimension IDs |  |
| if( vps\_nuh\_layer\_id\_present\_flag ) |  |
| **layer\_id\_in\_nuh**[ i ] | u(6) |
| **~~num\_dimensions\_minus1~~**~~[ i ]~~ | ~~u(4)~~ |
| for( j = 0; j <= num\_dimensions\_minus1; j++ ) ~~{~~ |  |
| **~~dimension\_type~~**~~[ i ][ j ]~~ | ~~u(4)~~ |
| **dimension\_id**[ i ][ j ] | u(v~~8~~) |
| ~~}~~ |  |
| } |  |
| for( i = 1; i <= vps\_max\_layers\_minus1 ; i++ ) |  |
| profile\_tier\_level( 1, vps\_max\_sub\_layers\_minus1 ) |  |
| for( i = 1; i <= vps\_max\_layers\_minus1; i++ ) { |  |
| // layer dependency |  |
| **num\_direct\_ref\_layers**[ i ] | u(6) |
| for( j = 0; j < num\_direct\_ref\_layers[ i ]; j++ ) |  |
| **ref\_layer\_id**[ i ][ j ] | u(6) |
| } |  |
| } |  |

[Semantics to be updated]

**vps\_extension\_byte\_alignment\_reserved\_one\_bit** shall be equal to 1.

**avc\_base\_layer\_flag** equal to 1 specifies that the base layer conforms to Rec. ITU-T H.264 | ISO/IEC 14496-10, equal to 0 specifies that it conforms to this specification.

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

**scalability\_mask** signals a pattern of 0 and 1 bits with each bit corresponding to one scalability dimension as indicated by the table below. A value of 1 for a particular scalability dimension indicates that this scalability dimension is present. A value of 0 for a particular scalability dimension indicates that this scalability dimension is not present. The values of NumScalabilityTypes is equal to the sum of number of bits in the scalability\_mask having value of 1. Thus



|  |  |
| --- | --- |
| **scalability\_mask** | **Scalability dimension** |
| 0 | none (base HEVC) |
| 1 | spatial |
| 2 | quality |
| 3 | depth |
| 4 | multiview |
| 5 | unspecified |
| 6-15 | Reserved |

**dimension\_id**[ i ][ j ] specifies the identifier of the j-th scalability dimension type of the i-th layer. When not present, the value of dimension\_id[ i ][ j ] is inferred to be equal to 0.

**num\_direct\_ref\_layers**[ i ] specifies the number of layers the i-th layer directly depends on. [Ed.(YK): Add the exact meaning of a layer directly depending on another layer.]

**ref\_layer\_id**[ i ][ j ] identifies the j-th layer the i-th layer directly depends on.

## Sequence parameter set syntax and semantics

Additions and removals in relative to the latest HEVC draft are highlighted and strikethrough, respectively (note that this would be the SPS syntax in the extension spec, not the one in the base spec, thus it is OK to use "nuh\_layer\_id" instead of "nuh\_reserved\_zero\_6bits").

|  |  |
| --- | --- |
| seq\_parameter\_set\_rbsp( ) { | Descriptor |
| **video\_parameter\_set\_id** | u(4) |
| **sps\_max\_sub\_layers\_minus1** | u(3) |
| **sps\_reserved\_zero\_bit** | u(1) |
| if (nuh\_layer\_id == 0) |  |
| profile\_tier\_level( 1, sps\_max\_sub\_layers\_minus1 ) |  |
| **seq\_parameter\_set\_id** | ue(v) |
| **chroma\_format\_idc** | ue(v) |
| if( chroma\_format\_idc = = 3 ) |  |
| **separate\_colour\_plane\_flag** | u(1) |
| **pic\_width\_in\_luma\_samples** | ue(v) |
| **pic\_height\_in\_luma\_samples** | ue(v) |
| **pic\_cropping\_flag** | u(1) |
| if( pic\_cropping\_flag ) { |  |
| **pic\_crop\_left\_offset** | ue(v) |
| **pic\_crop\_right\_offset** | ue(v) |
| **pic\_crop\_top\_offset** | ue(v) |
| **pic\_crop\_bottom\_offset** | ue(v) |
| } |  |
| **bit\_depth\_luma\_minus8** | ue(v) |
| **bit\_depth\_chroma\_minus8** | ue(v) |
| [Ed. (BB): chroma bit depth present in HM software but not used further ] |  |
| **pcm\_enabled\_flag** | u(1) |
| if( pcm\_enabled\_flag ) { |  |
| **pcm\_sample\_bit\_depth\_luma\_minus1** | u(4) |
| **pcm\_sample\_bit\_depth\_chroma\_minus1** | u(4) |
| } |  |
| **log2\_max\_pic\_order\_cnt\_lsb\_minus4** | ue(v) |
| for( i = 0; i <= sps\_max\_sub\_layers\_minus1; i++ ) { |  |
| **sps\_max\_dec\_pic\_buffering**[ i ] | ue(v) |
| **sps\_max\_num\_reorder\_pics**[ i ] | ue(v) |
| **sps\_max\_latency\_increase**[ i ] | ue(v) |
| } |  |
| **restricted\_ref\_pic\_lists\_flag** | u(1) |
| if( restricted\_ref\_pic\_lists\_flag ) |  |
| **lists\_modification\_present\_flag** | u(1) |
| **log2\_min\_luma\_coding\_block\_size\_minus3** | ue(v) |
| **log2\_diff\_max\_min\_luma\_coding\_block\_size** | ue(v) |
| **log2\_min\_transform\_block\_size\_minus2** | ue(v) |
| **log2\_diff\_max\_min\_transform\_block\_size** | ue(v) |
| if( pcm\_enabled\_flag ) { |  |
| **log2\_min\_pcm\_luma\_coding\_block\_size\_minus3** | ue(v) |
| **log2\_diff\_max\_min\_pcm\_luma\_coding\_block\_size** | ue(v) |
| } |  |
| **max\_transform\_hierarchy\_depth\_inter** | ue(v) |
| **max\_transform\_hierarchy\_depth\_intra** | ue(v) |
| **scaling\_list\_enable\_flag** | u(1) |
| if( scaling\_list\_enable\_flag ) { |  |
| **sps\_scaling\_list\_data\_present\_flag** | u(1) |
| if( sps\_scaling\_list\_data\_present\_flag ) |  |
| scaling\_list\_data( ) |  |
| } |  |
| **amp\_enabled\_flag** | u(1) |
| **sample\_adaptive\_offset\_enabled\_flag** | u(1) |
| if( pcm\_enabled\_flag ) |  |
| **pcm\_loop\_filter\_disable\_flag** | u(1) |
| **sps\_temporal\_id\_nesting\_flag** | u(1) |
| **num\_short\_term\_ref\_pic\_sets** | ue(v) |
| for( i = 0; i < num\_short\_term\_ref\_pic\_sets; i++) |  |
| short\_term\_ref\_pic\_set( i ) |  |
| **long\_term\_ref\_pics\_present\_flag** | u(1) |
| if( long\_term\_ref\_pics\_present\_flag ) { |  |
| **num\_long\_term\_ref\_pics\_sps** | ue(v) |
| for( i = 0; i < num\_long\_term\_ref\_pics\_sps; i++ ) { |  |
| **lt\_ref\_pic\_poc\_lsb\_sps**[ i ] | u(v) |
| **used\_by\_curr\_pic\_lt\_sps\_flag**[ i ] | u(1) |
| } |  |
| } |  |
| **sps\_temporal\_mvp\_enable\_flag** | u(1) |
| **vui\_parameters\_present\_flag** | u(1) |
| if( vui\_parameters\_present\_flag ) |  |
| vui\_parameters( ) |  |
| **sps\_extension\_flag** | u(1) |
| if( sps\_extension\_flag ) |  |
| while( more\_rbsp\_data( ) ) |  |
| **sps\_extension\_data\_flag** | u(1) |
| rbsp\_trailing\_bits( ) |  |
| } |  |

# Items for further study

Further study is encouraged on the following items.

1. Efficient signaling of profile\_tier\_level in the vps\_extension( )
2. Options for signaling an additional, possibly more restrictive level limit in the SPS, and consideration if the level limit can apply to an individual layer rather than an operation point
3. External means for signaling information otherwise found in the vps\_extension( )
4. Reduction of SPS related redundancy across layers by using inter-layer SPS prediction or putting common information into the VPS
5. Further partitioning of the scalability “dimension\_ids” into separate components