|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  9th Meeting: Geneva, CH, 27 April – 7 May 2012 | Document: JCTVC-I0571 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **AHG12: Video parameter set and its use in 3D-HEVC** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Ye-Kui Wang Ying Chen Marta Karczewicz  5775 Morehouse Dr San Diego, CA 92121 USA | Tel: Email: | 1-858-651-8345 [yekuiw@qualcomm.com](mailto:yekuiw@qualcomm.com)  1-858-845-6589 [cheny@qualcomm.com](mailto:cheny@qualcomm.com) |
| *Source:* | Qualcomm Incorporated | | |

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

# Abstract

This document proposes the inclusion of the video parameter set (VPS), which is one level above the SPS. The syntax of the VPS, including its extension in the HEVC based 3DV is shown, wherein 1) the view\_id of each view is signalled in the sequence level, similar to MVC; 2) the view dependency of each view is signalled in the sequence level; and 3) a video parameter set can be shared by multiple SPSs.

# The video parameter set design

## Syntax elements for view dependency

For the view dependency, it is proposed that dependent views that are used for different types of inter-view prediction (besides inter-view texture prediction) are jointly signalled and a flag is introduced for the views used for inter-view texture prediction.

### Syntax

|  |  |
| --- | --- |
| view\_dependency( ) { |  |
| **num\_views\_minus1** | ue(v) |
| for( i = 0; i <= num\_views\_minus1; i++ ) |  |
| **view\_id[**i **]** | ue(v) |
| for( i = 1; i <= num\_views\_minus1; i++ ) { |  |
| **num\_ref\_views[** i **]** |  |
| for( j = 0; j < num\_ref\_views[ i ]; j++ ) |  |
| **ref\_view\_idx[** i **][** j **]** | ue(v) |
| **inter\_view\_texture\_flag[** i **][** j **]** | u(1) |
| } |  |
| } |  |
| } |  |

### Semantics

**num\_views\_minus1** plus 1 specifies the maximum number of coded views in the coded video sequence. The value of num\_view\_minus1 shall be in the range of 0 to 30, inclusive.

**num\_ref\_views[** i **]** specifies the number of dependent views to be used to decode the view with view\_id equal view\_id[ i ] and with view index equal to i. The value of num\_ref\_views[ i ] shall not be greater than Min( 15, num\_views\_minus1 ). The value of num\_ref\_views[ 0 ] shall be equal to 0.

**ref\_view\_idx[** i **][** j **]** specifies the view index of the j-th dependent view when decoding a view component with view index equal to i. The value of ref\_view\_idx[ i ][ j ] shall be in the range of 0 to 30, inclusive.

**inter\_view\_texture\_flag[** i **][** j **]** equal to 1 specifies that the view component with view index equal to ref\_view\_idx[ i ][ j ] is to be included in the inter-view RPS subset to be used for reference picutre list construction of the view components with view index equal to i. inter\_view\_texture\_flag[ i ][ j ] equal to 0 specifies that the view component with view index equal to ref\_view\_idx[ i ][ j ] is not included in the inter-view RPS subset to be used for reference picutre list construction. The view componets added into the inter-view RPS subset for the decoding of a view component with view index equal to i, are in the assending order of j.

## Presence of the above syntax elements in the bitstream

This part of proposal requires modification of the HEVC specification. In this proposal, a video parameter set is proposed and sequence parameter sets refer to the video parameter set. The above syntax elements are added in the proposed video parameter set, which is similar to that video parameter set concept proposed in [2] with some modifications. Note that if video parameter set is not introduced, the proposed syntax elements can be in the extension of the sequence parameter set.

### Video parameter set syntax

The highlighted syntax elements are proposed to be included in the VPS in the HEVC base spec.

|  |  |
| --- | --- |
| video\_parameter\_set\_rbsp( ) { | Descriptor |
| **video\_parameter\_set\_id** | ue(v) |
| **num\_temporal\_layers\_minus1** | u(3) |
| for ( i = 0; i <= num\_temporal\_layers\_minus1; i++ ) { |  |
| **profile\_idc**[ i ] | u(8) |
| **reserved\_zero\_8bits**[ i ] /\* equal to 0 \***/** | u(8) |
| **level\_idc**[ i ] | u(8) |
| } |  |
| **bit\_depth\_luma\_minus8** | ue(v) |
| **bit\_depth\_chroma\_minus8** | ue(v) |
| **chroma\_format\_idc** | u(2) |
| **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) |
| } |  |
| **temporal\_id\_nesting\_flag** | u(1) |
| **bit\_equal\_to\_one** /\* equal to 1 \*/ | u(1) |
| **extension\_type** /\* equal to 0 for 3DV \*/ | ue(v) |
| **num\_layers\_minus2** | ue(v) |
| **num\_rep\_formats\_minus1** | ue(v) |
| for( i = 1; i <= num\_rep\_formats\_minus1; i++ ) { |  |
| **bit\_depth\_luma\_minus8[** i **]** | ue(v) |
| **bit\_depth\_chroma\_minus8[** i **]** | ue(v) |
| **chroma\_format\_idc[** i **]** | u(2) |
| **pic\_width\_in\_luma\_samples[** i **]** | ue(v) |
| **pic\_height\_in\_luma\_samples[** i **]** | ue(v) |
| **pic\_cropping\_flag[** i **]** | u(1) |
| if( pic\_cropping\_flag[ i ] ) { |  |
| **pic\_crop\_left\_offset[** i **]** | ue(v) |
| **pic\_crop\_right\_offset[** i **]** | ue(v) |
| **pic\_crop\_top\_offset[** i **]** | ue(v) |
| **pic\_crop\_bottom\_offset[** i **]** | ue(v) |
| } |  |
| } |  |
| for( i = 1; i <= num\_layers\_minus1; i++ ) { |  |
| **rep\_format\_idx[** i **]** | ue(v) |
| if( extension\_type = = 1) { // This condition is always not true for 3DV |  |
| **dependency\_id**[ i ] | ue(v) |
| **quality\_id**[ i ] | ue(v) |
| **num\_directly\_dependent\_layers[** i **]** | ue(v) |
| for( j = 0; j < num\_directly\_dependency\_layers[ i ]; j++ ) |  |
| **delta**\_**reference\_layer\_id\_minus1**[ i ][ j ] | ue(v) |
| } |  |
| } |  |
| **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 ) |  |
| if ( extension\_type = =0 ) |  |
| view\_dependency( ) |  |
| **num\_additional\_profiles\_levels\_minus1** | ue(v) |
| for( i = 0; i <= num\_additional\_profiles\_levels\_minus1; i++ ) { |  |
| **additional\_profile\_idc[** i **]** | u(8) |
| **additional\_reserved\_zero\_8bits[** i **]** /\* equal to 0 \***/** | u(8) |
| **additional\_level\_idc[** i **]** | u(8) |
| **num\_applicable\_operation\_points\_minus1[** i **]** | ue(v) |
| for( j = 0; j <= num\_applicable\_operation\_points[ i ]; j++ ) { |  |
| **temporal\_id[** i **][** j **]** | ue(v) |
| if( extension\_type = = 0 ) { /\* Always true for 3DV \*/ |  |
| **depth\_included\_flag** | u(1) |
| **num\_target\_output\_views\_minus1[** i **][** j **]** | ue(v) |
| for( k = 0; k < num\_target\_output\_views\_minus1[ i ][ j ]; k++ ) |  |
| **layer\_id[** i **][** j **][** k **]** | ue(v) |
| } |  |
| else (extension\_type = = 1) // This condition is always not   true for 3DV |  |
| **layer\_id[** i **][** j **]** | ue(v) |
| } |  |
| } |  |
| **vps\_vui\_parameters\_present\_flag** | u(1) |
| if( vps\_vui\_parameters\_present\_flag ) |  |
| vps\_vui\_parameters( ) |  |
| **vps\_extension\_flag** | u(1) |
| if( vps\_extension\_flag ) |  |
| while( more\_rbsp\_data( ) ) |  |
| **vps\_extension\_data\_flag** | u(1) |
| rbsp\_trailing\_bits( ) |  |
| } |  |

### Video parameter set semantics

*Note that the semantics related to the scalable extension of HEVC is not given in this proposal although the above syntax also supports the scalable extension of HEVC.*

**video\_parameter\_set\_id** specifies the identifier of the current video parameter set.

**num\_temporal\_layers\_minus1, profile\_idc**[ i ], **reserved\_zero\_8bits**[ i ], **level\_idc**[ i ], **bit\_depth\_luma\_minus8, bit\_depth\_chroma\_minus8, chroma\_format\_idc, pic\_width\_in\_luma\_samples, pic\_height\_in\_luma\_samples, pic\_cropping\_flag, pic\_crop\_left\_offset, pic\_crop\_right\_offset, pic\_crop\_top\_offset, pic\_crop\_bottom\_offset,** and **temporal\_id\_nesting\_flag** have the same semantics of those syntax elements of sequence parameter set in the current HEVC WD with the same names.

**bit\_equal\_to\_one** shall be equal to 1.

**extention\_type** equal to 0 that multiple view layers may be present in the bitstream.

**num\_rep\_formats\_minus1** plus 1 specifies the maximum number of different sets representation formats supported by this video parameter set, a representation format include bit depth and chroma format (i.e., the sets of bit\_depth\_luma\_minus8, bit\_depth\_chroma\_minus8, and chroma\_format\_idc values) , picture resolution and cropping window information in the coded video sequence. The value of num\_rep\_formats\_minus1 shall be in the range of 0 to X, inclusive. The set of bit depth and chroma format for the base layer is signalled by bit\_depth\_luma\_minus8, bit\_depth\_chroma\_minus8, and chroma\_format\_idc, and sets of bit depth and chroma format are signalled for enhancement layers by the following set of syntax elements bit\_depth\_luma\_minus8[ i ], bit\_depth\_chroma\_minus8[ i ], and chroma\_format\_idc[ i ].

The first set (with index 0) of representation format is signalled by bit\_depth\_luma\_minus8, bit\_depth\_chroma\_minus8, chroma\_format\_idc, pic\_width\_in\_luma\_samples, pic\_height\_in\_luma\_samples, pic\_cropping\_flag, pic\_crop\_left\_offset, pic\_crop\_right\_offset, pic\_crop\_top\_offset, and pic\_crop\_bottom\_offset.

**bit\_depth\_luma\_minus8[** i **]**, **bit\_depth\_chroma\_minus8[** i **]**, and **chroma\_format\_idc[** i **]** specify the i-th set of bit\_depth\_luma\_minus8, bit\_depth\_chroma\_minus8, and chroma\_format\_idc values in the coded video sequence.

**pic\_width\_in\_luma\_samples[** i **] and pic\_height\_in\_luma\_samples[** i **]** specify the height and width of each decoded picture in units of luma samples using the i-th representation format.

**pic\_cropping\_flag[** i **]** **pic\_crop\_left\_offset[** i **]**, **pic\_crop\_right\_offset[** i **]**, **pic\_crop\_top\_offset[** i **]**, and **pic\_crop\_bottom\_offset[** i **]** specify for the i-th set of representation format, the samples of the pictures in the coded video sequence that are output from the decoding process, in terms of a rectangular region specified in picture coordinates for output.

**rep\_format\_idx[** i **]** specifies the values index to the set of additional bit depth and chorma format that applies to the layer with layer\_id equal to i. The values of bit\_depth\_luma\_minus8, bit\_depth\_chroma\_minus8, and chroma\_format\_idc for the layer with layer\_id equal to i are equal to bit\_depth\_luma\_minus8[ rep\_format\_idx[ i ] ], bit\_depth\_chroma\_minus8[ rep\_format\_idx[ i ] ], and chroma\_format\_idc[ rep\_format\_idx[ i ] ], respectively. The value of rep\_format\_idx[ i ] shall be in the range of 0 to X, inclusive.

**num\_short\_term\_ref\_pic\_sets** specifies the number of short-term reference picture sets that are specified in the video parameter set. The value of num\_short\_term\_ref\_pic\_sets shall be in the range of 0 to 64, inclusive.

**num\_additional\_profiles\_levels\_minus1** plus 1specifies the number of additional set of profile and level information signalled in the video parameter set. The value of num\_additional\_profiles\_levels\_minus1 shall be in the range of 0 to X, inclusive.

**additional\_profile\_idc[** i **]**, **additional\_reserved\_zero\_8bits[** i **]**, and **additional\_level\_idc[** i **]** specify the i-th additional set of profile and level information signalled in the video parameter set.

**num\_applicable\_operation\_points\_minus1[** i **]** plus 1 specifies the number of operation points to which the i-th set of additional profile and level information applies.

**temporal\_id[** i **][** j **]** specifies the temporal\_id of the j-th operation point to which the i-th additional set of profile and level information applies.

**num\_target\_output\_views\_minus1[** i **][** j **]** plus 1 specifies the number of target output views in the j-th operation point to which the i-th additional set of profile and level information applies.

**depth\_included\_flag** equal to 1 indicates that the current 3DV operation point contains depth. depth\_included\_flag equal to 0 indicates that the current 3DV operation point does not contain depth.

**layer\_id[** i **][** j **][** k **]** specifies the layer identifier of the k-th target output view of the j-the operation point to which the i-th additional set of profile and level information applies.

### Sequence parameter set syntax

|  |  |
| --- | --- |
| seq\_parameter\_set\_rbsp( ) { | Descriptor |
| **~~profile\_idc~~** | ~~u(8)~~ |
| **~~reserved\_zero\_8bits~~** ~~/\* equal to 0 \*~~**~~/~~** | ~~u(8)~~ |
| **~~level\_idc~~** | ~~u(8)~~ |
| **seq\_parameter\_set\_id** | ue(v) |
| **~~max\_temporal\_layers\_minus1~~** | ~~u(3)~~ |
| **~~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)~~ |
| **video\_parameter\_set\_id** | ue(v) |
| **rep\_format\_idx** | ue(v) |
| **pcm\_enabled\_flag** | u(1) |
| if( pcm\_enabled\_flag ) { |  |
| **pcm\_bit\_depth\_luma\_minus1** | u(4) |
| **pcm\_bit\_depth\_chroma\_minus1** | u(4) |
| } |  |
| **qpprime\_y\_zero\_transquant\_bypass\_flag** | u(1) |
| **log2\_max\_pic\_order\_cnt\_lsb\_minus4** | ue(v) |
| for( i = 0; i <= max\_temporal\_layers\_minus1; i++ ) { |  |
| **max\_dec\_pic\_buffering[** i **]** | ue(v) |
| **num\_reorder\_pics[** i **]** | ue(v) |
| **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\_coding\_block\_size\_minus3** | ue(v) |
| **log2\_diff\_max\_min\_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\_coding\_block\_size\_minus3** | ue(v) |
| **log2\_diff\_max\_min\_pcm\_coding\_block\_size** | ue(v) |
| } |  |
| **max\_transform\_hierarchy\_depth\_inter** | ue(v) |
| **max\_transform\_hierarchy\_depth\_intra** | ue(v) |
| **scaling\_list\_enable\_flag** |  |
| **chroma\_pred\_from\_luma\_enabled\_flag** | u(1) |
| **deblocking\_filter\_in\_aps\_enabled\_flag** | u(1) |
| **seq\_loop\_filter\_across\_slices\_enabled\_flag** | u(1) |
| **asymmetric\_motion\_partitions\_enabled\_flag** | u(1) |
| **non\_square\_quadtree\_enabled\_flag** | u(1) |
| **sample\_adaptive\_offset\_enabled\_flag** | u(1) |
| **adaptive\_loop\_filter\_enabled\_flag** | u(1) |
| if( adaptive\_loop\_filter\_enabled\_flag ) |  |
| **alf\_coef\_in\_slice\_flag** | u(1) |
| if( pcm\_enabled\_flag ) |  |
| **pcm\_loop\_filter\_disable\_flag** | u(1) |
| **temporal\_id\_nesting\_flag** | u(1) |
| if( log2\_min\_coding\_block\_size\_minus3 = = 0 ) |  |
| **inter\_4x4\_enabled\_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) |
| **tiles\_or\_entropy\_coding\_sync\_idc** | u(2) |
| if( tiles\_or\_entropy\_coding\_sync\_idc = = 1 ) { |  |
| **num\_tile\_columns\_minus1** | ue(v) |
| **num\_tile\_rows\_minus1** | ue(v) |
| **uniform\_spacing\_flag** | u(1) |
| if( !uniform\_spacing\_flag ) { |  |
| for( i = 0; i < num\_tile\_columns\_minus1; i++ ) |  |
| **column\_width[**i **]** | ue(v) |
| for( i = 0; i < num\_tile\_rows\_minus1; i++ ) |  |
| **row\_height[**i **]** | ue(v) |
| } |  |
| **loop\_filter\_across\_tiles\_enabled\_flag** | u(1) |
| } |  |
| **vui\_parameters\_present\_flag** | u(1) |
| if( vui\_parameters\_present\_flag ) |  |
| vui\_parameters( ) |  |
| **sps\_extension\_flag** | u(1) |
| if( sps\_extension\_flag ) |  |
| while( more\_rbsp\_data( ) ) |  |
| **sps\_extension\_data\_flag** | u(1) |
| rbsp\_trailing\_bits( ) |  |
| } |  |

### Sequence parameter set semantics

**video\_parameter\_set\_id** identifies the video parameter set (VPS) referred by the current SPS.

**rep\_format\_idx** specifies the index to the representation format signalled in the referred video parameter set.

# Patent rights declaration(s)

**Qualcomm Incorporated may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard (per box 2 of the ITU-T/ITU-R/ISO/IEC patent statement and licensing declaration form).**