|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  14th Meeting: Vienna, AT, 25 July – 2 Aug. 2013 | Document: JCTVC-N0109r1 |

|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extensions**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  5th Meeting: Vienna, AT, 27 July – 2 Aug. 2013 | Document: JCT3V-E0067 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **MV-HEVC/SHVC HLS: Signalling for sub-layer dependency** | | |
| *Status:* | Input Document to JCT-VC and JCT-3V | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Vadim Seregin Ye-Kui Wang  5775 Morehouse Drive San Diego, CA 92121-1714 USA | Email: | [vseregin@qti.qualcomm.com](mailto:vseregin@qti.qualcomm.com) [yekuiw@qti.qualcomm.com](mailto:yekuiw@qualcomm.com) |
| *Source:* | Qualcomm Incorporated | | |

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

# Abstract

This contribution proposes to further classify the sequence-level inter-layer dependency, which is currently derived from the syntax element direct\_dependency\_flag[ i ][ j ], to be sub-layer specific, by utilizing information carried by the syntax element max\_sublayer\_for\_ilp\_plus1[ i ]. Specifically, the value of NumDirectRefLayers that is currently defined for each layer is defined for each sub-layer of each layer, thus becoming a two-dimension array. The sub-layer classified variable is then applied in reference picture marking and picture-level inter-layer reference picture signalling, in order to mark certain pictures as "unused for reference" earlier and release the picture buffer for storing other decoded pictures or to avoid sending of unnecessary bits in slice headers for signaling of pictures used for inter-layer prediction by each picture.

# Sub-layer dependency

In the current VPS design, there are two syntax elements, direct\_dependency\_flag and max\_sublayer\_for\_ilp\_plus1, that define reference layer dependency. Depending on the value of max\_sublayer\_for\_ilp\_plus1, the number of reference layers for a particular sub-layer can be smaller than what is specified in VPS with NumDirectRefLayers.

The current derivation of NumDirectRefLayers does not take into account max\_sublayer\_for\_ilp\_plus1 information and thus is not optimal for certain sub-layers. For example, there could be two reference layers specified by direct\_dependency\_flag and only one can be allowed by max\_sublayer\_for\_ilp\_plus1 for a certain temporal ID. In this case there is no need to further signal the reference layer ID in a slice header for that sub-layer.

In this contribution, it is proposed to include the information carried by the max\_sublayer\_for\_ilp\_plus1 in derivation of NumDirectRefLayers making it a two-dimensional array.

The sub-layer classified variable is then applied in reference picture marking and picture-level inter-layer reference picture signalling, thus to mark certain pictures as "unused for reference" earlier and release the picture buffer for storing other decoded pictures or to avoid sending of unnecessary bits in slice headers for signaling of pictures used for inter-layer prediction by each picture.

# Specification change

This section provides the examples of the specification changes associated with the inclusion of the max\_sublayer\_for\_ilp\_plus1 in the NumDirectRefLayers derivation. These changes are marked with magenta colour.

## Including temporal Id for NumDirectRefLayers

To optimise the usage, it is suggested to include the temporal Id dependency in the derivation of the NumDirectRefLayers. It can be seen as adding one more dimension to this variable. Basically, the direct\_dependency\_flag and max\_sublayer\_for\_ilp\_plus1 will be used together to define a NumDirectRefLayers variable.

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

for( i = 0; i < 64; i++ ) {  
 NumSamplePredRefLayers[ i ] = 0  
 NumMotionPredRefLayers[ i ] = 0

for( k = 0; k < 8; k++)  
 NumDirectRefLayers[ i ][ k ] = 0  
 for( j = 0; j < 64; j++ ) {  
 SamplePredEnabledFlag[ i ][ j ] = 0  
 MotionPredEnabledFlag[ i ][ j ] = 0  
 RefLayerId[ i ][ j ] = 0  
 SamplePredRefLayerId[ i ][ j ] = 0  
 MotionPredRefLayerId[ i ][ j ] = 0  
 }  
}

for( i = 1; i <= vps\_max\_layers\_minus1; i++ ) {  
 iNuhLId = layer\_id\_in\_nuh[ i ]  
 for( j = 0; j < i; j++ )  
 if( direct\_dependency\_flag[ i ][ j ] ) {

if(max\_tid\_il\_ref\_pics\_plus1[ j ] = = 0 )  
 RefLayerId[ iNuhLId ][ NumDirectRefLayers[ iNuhLId ][0]++ ] = layer\_id\_in\_nuh[ j ]

else

for( k = 0; k < max\_tid\_il\_ref\_pics\_plus1[ j ]; k++ )

RefLayerId[ iNuhLId ][ NumDirectRefLayers[ iNuhLId ][ k ]++ ] = layer\_id\_in\_nuh[ j ]

SamplePredEnabledFlag[ iNuhLId ][ j ] = ( ( direct\_dependency\_type[ i ][ j ] + 1 ) & 1 )  
 NumSamplePredRefLayers[ iNuhLId ] += SamplePredEnabledFlag[ iNuhLId ][ j ]  
 MotionPredEnabledFlag[ iNuhLId ][ j ] = ( ( ( direct\_dependency\_type[ i ][ j ] + 1 ) & 2 ) >> 1 )  
 NumMotionPredRefLayers[ iNuhLId ] += MotionPredEnabledFlag[ iNuhLId ][ j ]  
 }

}

When the NumDirectRefLayers check is necessary the former usage should be extended by including the templral Id. For example, in slice header signaling it can be achieved by the following change.

|  |  |
| --- | --- |
| slice\_segment\_header( ) { | Descriptor |
| … |  |
| if( nuh\_layer\_id > 0 && NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] > 0 ) { |  |
| **inter\_layer\_pred\_enabled\_flag** | u(1) |
| if( inter\_layer\_pred\_enabled\_flag &&  NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] > 1) { |  |
| if( !max\_one\_active\_ref\_layer\_flag ) { |  |
| **num\_inter\_layer\_ref\_pics\_minus1** | u(v) |
| NumActiveRefLayerPics = num\_inter\_layer\_ref\_pics\_minus1 + 1 |  |
| } else |  |
| NumActiveRefLayerPics = 1 |  |
| for( i = 0; i < NumActiveRefLayerPics; i++ ) |  |
| **inter\_layer\_pred\_layer\_idc[**i ] | u(v) |
| } else |  |
| NumActiveRefLayerPics = inter\_layer\_pred\_enabled\_flag ? 1 : 0 |  |
| } else |  |
| NumActiveRefLayerPics = 0 |  |
| if( NumSamplePredRefLayers[ nuh\_layer\_id ] > 0 && NumActiveRefLayerPics > 0 ) |  |
| **inter\_layer\_sample\_pred\_only\_flag** | u(1) |
| if( slice\_segment\_header\_extension\_present\_flag ) { |  |
| **slice\_segment\_header\_extension\_length** | ue(v) |
| for( i = 0; i < slice\_segment\_header\_extension\_length; i++) |  |
| **slice\_segment\_header\_extension\_data\_byte**[ i ] | u(8) |
| } |  |
| byte\_alignment( ) |  |
| } |  |

And as can be seen, this change provides more optimal signaling for other syntax elements. For example, inter\_layer\_pred\_enabled\_flag is not signaled if there is no dependency layer for current sub-layer specified by TemporalId. Without this change, this syntax element is required to be signaled but shall be set equal to 0 for a sub-layer without inter-layer prediction.

Similar modification can be done for other places where NumDirectRefLayers is required.

## When former value of NumDirectRefLayers is required

In some places, where Temporal Id is not available, the NumDirectRefLayers checking can be done setting the TemporalId equal to 0. For example, in layer dependency change SEI message syntax it can be done as follows.

|  |  |
| --- | --- |
| layer\_dependency\_change( payloadSize ) { | **Descriptor** |
| **active\_vps\_id** | u(4) |
| for( i = 1; i <= vps\_max\_layers\_minus1; i++ ) |  |
| for( j = 0; j < NumDirectRefLayers[ layer\_id\_in\_nuh[ i ] ][ 0 ]; j++ ) |  |
| **ref\_layer\_disable\_flag**[ i ][ j ] | u(1) |
| } |  |

TemporalId equal to 0 in NumDirectRefLayers defines the lowest possible reference layer dependency, in other words if direct\_dependency\_flag[ currentLayerId ][ referenceLayerId ] is equal to 1 for some particular reference layer with nuh\_layer\_id equal to referenceLayerId, at least NumDirectRefLayers[currentLayerId][0] should be greater than 0.

## Using temporalId for picture marking

Finally, in a similar way, temporal Id checking for NumDirectRefLayers can be involved in the picture marking process and can be implemented as follows.

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

Input to this process is:

– a nuh\_layer\_id value latestDecLayerId

Output of this process is:

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

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

The variables numTargetDecLayers, and latestDecIdx are derived as follows:

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

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

The following applies for marking of pictures as "unused for reference":

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

– Let currPic be the picture in the current access unit with nuh\_layer\_id equal to TargetDecLayerIdList[ i ].

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

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

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

remainingInterLayerReferencesFlag = 0  
 if ( currTid <= ( max\_sublayer\_for\_ilp\_plus1[ LayerIdInVps[ TargetDecLayerIdList[ i ] ] ] –1 ) )   
 for( j = latestDecIdx + 1; j < numTargetDecLayers; j++ )  
 for( k = 0; k < NumDirectRefLayers[ TargetDecLayerIdList[ j ] ][ currTid ]; k++ )  
 if( TargetDecLayerIdList[ i ] = = RefLayerId[ TargetDecLayerIdList[ j ] ][ k ] )  
 remainingInterLayerReferencesFlag = 1

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

## Related semantic changes

**layer\_present\_flag**[ i ]equal to 1 indicates that there may or may not be NAL units in the target access units with nuh\_layer\_id  equal to layer\_id\_in\_nuh[ i ]. layer\_present\_flag[ i ]equal to 0 indicates that there are no NAL units in the target access units with nuh\_layer\_id equal to layer\_id\_in\_nuh[ i ]. [Ed. (YK): The definition of "the target access units" is missing.]

When layer\_present\_flag[ i ]is equal to 1 and i is greater than 0, layer\_present\_flag[ LayerIdxInVps[ RefLayerId[ layer\_id\_in\_nuh[ i ] ][ j ] ] ] shall be equal to 1 for all values of j in the range of 0 to NumDirectRefLayers[ layer\_id\_in\_nuh[ i ] ][ 0 ] − 1, inclusive.

**num\_inter\_layer\_ref\_pics\_minus1** plus 1 specifies the number of pictures that may be used in decoding of the current picture for inter-layer prediction. The length of the num\_inter\_layer\_ref\_pics\_minus1 syntax element is Ceil( Log2( NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] ) ) bits. The value of num\_inter\_layer\_ref\_pics\_minus1 shall be in the range of 0 to NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] − 1, inclusive.

The variable NumActiveRefLayerPics is derived as follows:

if( nuh\_layer\_id = = 0 | | NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] = = 0 | | !inter\_layer\_pred\_enabled\_flag )  
 NumActiveRefLayerPics = 0  
else if( max\_one\_active\_ref\_layer\_flag | | NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] = = 1 )  
 NumActiveRefLayerPics = 1  
else  
 NumActiveRefLayerPics = num\_inter\_layer\_ref\_pics\_minus1 + 1

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

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

# Analysis of bit saving and DPB size saving

The maximum bit saving can be achieved when inter-layer prediction is used only for IRAP pictures or only for pictures with TemporalId equal to 0, i.e. when max\_tid\_il\_ref\_pics\_plus1[ i ] for all i values is equal to 0 or 1 for all layers, which can be up to 64 layers.

Next table summarizes bit saving for each AU (assuming that each picture is coded as one slice) in multi-layer coding, assuming that all lower layers are dependent by the higher layers.

|  |  |
| --- | --- |
| Number of layers | Bits saving |
| 2 | 1 |
| 3 | 4 |
| 4 | 12 |
| 5 | 22 |
| 6 | 40 |
| 7 | 61 |
| 8 | 85 |
| … | … |
| 64 | 11698 |

The total bit savings can be up to 11698 bits per AU, assuming all pictures contain only one slice, with TemporalId greater than 0 in the top layer when the number of layer is 64. The calculation method is shown based on the syntax table below.

|  |  |  |
| --- | --- | --- |
| slice\_segment\_header( ) { | Descriptor | Max bit saving |
| … |  |  |
| if( nuh\_layer\_id > 0 && NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] > 0 ) { |  |  |
| **inter\_layer\_pred\_enabled\_flag** | u(1) | **1** |
| if( inter\_layer\_pred\_enabled\_flag &&  NumDirectRefLayers[ nuh\_layer\_id ][ TemporalId ] > 1) { |  |  |
| if( !max\_one\_active\_ref\_layer\_flag ) { |  |  |
| **num\_inter\_layer\_ref\_pics\_minus1** | u(v) | **315** |
| NumActiveRefLayerPics = num\_inter\_layer\_ref\_pics\_minus1 + 1 |  |  |
| } else |  |  |
| NumActiveRefLayerPics = 1 |  |  |
| for( i = 0; i < NumActiveRefLayerPics; i++ ) |  |  |
| **inter\_layer\_pred\_layer\_idc[**i ] | u(v) | **11382** |
| } else |  |  |
| NumActiveRefLayerPics = inter\_layer\_pred\_enabled\_flag ? 1 : 0 |  |  |
| } else |  |  |
| NumActiveRefLayerPics = 0 |  |  |
| if( NumSamplePredRefLayers[ nuh\_layer\_id ] > 0 && NumActiveRefLayerPics > 0 ) |  |  |
| **inter\_layer\_sample\_pred\_only\_flag** | u(1) |  |
| if( slice\_segment\_header\_extension\_present\_flag ) { |  |  |
| **slice\_segment\_header\_extension\_length** | ue(v) |  |
| for( i = 0; i < slice\_segment\_header\_extension\_length; i++) |  |  |
| **slice\_segment\_header\_extension\_data\_byte**[ i ] | u(8) |  |
| } |  |  |
| byte\_alignment( ) |  |  |
| } |  |  |

With current proposal, there is no additional advantage on DPB size saving and only simplification in condition checking is achieved. To achieve the DPB size saving the RefLayerId needs to be modified to be also sub-layer specific.

# 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).**