|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extension Development**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  3rd Meeting: Geneva, CH, 17–23 Jan. 2013 | Document: JCT3V-C0049\_proposed text |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Proposed text for JCT3V-C0049 based on 3D-HEVC Test Model 2** | | |
| *Status:* | Input Document | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Li Zhang  Ying Chen Xiang Li Marta Karczewicz  5775 Morehouse Drive San Diego, CA 92121 USA | Tel: Email: | 1-858-651-6660 [lizhang@qti.qualcomm.com](mailto:lizhang@qti.qualcomm.com)  1-858-845-6589 [cheny@qti.qualcomm.com](mailto:cheny@qti.qualcomm.com) |
| *Source:* | Qualcomm Incorporated | | |

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

# Abstract

This document provides text for Draft 2 of 3D-HEVC Test Model Description. All the changes are highlighted in green.

|  |  |
| --- | --- |
| vps\_extension( ) { | Descriptor |
| … |  |
| for( i = 0; i <= vps\_max\_layers\_minus1; i++ ) { |  |
| if ( (i ! = 0)  && !( i % 2 ) ) { |  |
| **multi\_view\_mv\_pred\_flag**[ i ] | u(1) |
| **multi\_view\_residual\_pred\_flag**[ i ] | u(1) |
| } |  |
| … |  |
| } |  |

G.7.4.2.1.1 Video parameter set extension semantics

**multi\_view\_residual\_pred\_flag** [ i ]indicates whether inter-view residual prediction is used in the decoding process of the layer with layer\_id equal to i. multi\_view\_residual\_pred\_flag[ i ] equal to 0 specifies that inter-view residual prediction is not used for the layer with layer\_id equal to i. multi\_view\_residual\_pred\_flag[ i ]equal to 1 specifies that inter-view residual prediction may be used for the layer with layer\_id equal to i. When multi\_view**\_**residual\_pred\_flag [ i ] is not present, its value shall be inferred to be equal to 0.

G.7.3.9.1 General Coding unit syntax

|  |  |
| --- | --- |
| coding\_unit( x0, y0, log2CbSize , ctDepth) { | **Descriptor** |
| … |  |
| if ( resPredEnableFlag ) |  |
| ~~res\_pred\_flag~~ | ~~ae(v)~~ |
| **weighting\_factor\_index** | ae(v) |
| if ( icEnableFlag ) |  |
| **ic\_flag** | ae(v) |
| … |  |
| } |  |
| } |  |

**weighting\_factor\_index ~~res\_pred\_flag~~** equal to 0 specifies that residual prediction is not used. weighting\_factor\_index~~res\_pred\_flag~~unequal to ~~1~~ 0 specifies that residual prediction is used. When weighting\_factor\_index~~res\_pred\_flag~~ is not present, its value shall be inferred to be equal to 0.

~~The variable residualCbfNonZero is derived by values of cbf\_luma, cbf\_cb, cbf\_cr and PredMode of the corresponding blocks as follows: residualCbfNonZero is set to 1 if at least one of the corresponding blocks has both PredMode not equal to MODE\_INTRA and any of the values of cbf\_luma, cbf\_cb and cbf\_cr not equal to 0; otherwise, residualCbfNonZero is set equal to 0. The corresponding blocks are identified by the current PU and the disparity vector. All the corresponding blocks belong to transform units that are covered or partially covered by a corresponding rectangle area (of the current PU) in the inter-view reference view component, after shifting the PU location with a disparity vector.~~

The variable resPredEnableFlag specifying whether weighting\_factor\_index ~~res\_pred\_flag~~ is present in the bitstream is derived as

* 1. resPredEnableFlag = multi\_residual\_pred\_flag ~~&& residualCbfNonZero~~ && TempRefExistInSlice && PredMode != MODE\_INTRA && PartMode = = PART\_2Nx2N && anyTempRefPicFlag (G‑28)

1. The variable TempRefExistInSlice is derived at slice level and set to 1 if there is at least one reference picture from current view in either reference picture list. Otherwise, it is set to 0..

G.8.5.2.2 Decoding process for inter prediction samples

Inputs to this process are:

…

Outputs of this process are:

…

The variable nCSL is set equal to nCS and the variable nCSC is set equal to nCS >> 1.

When weighting\_factor\_index is unequal to 0, the derivation process for a disparity vector as specified in subclause G.8.5.2.1.13 is invoked with the luma locations ( xC, yC ) and ( xP, yP ), the coding block size nCS, the variables nPSW and nPSH, and the partition index 0 as the inputs and the outputs are the view order index refViewIdx, the flag availableDV and the disparity vector mvDisp.

Let predSamplesL0L and predSamplesL1L be (nPbW)x(nPbH) arrays of predicted luma sample values and predSampleL0Cb, predSampleL1Cb, predSampleL0Cr, and predSampleL1Cr be (nPbW/2)x(nPbH/2) arrays of predicted chroma sample values.

For LX being replaced by either L0 or L1 in the variables predFlagLX, RefPicListX, refIdxLX, refPicLX, and predPartLX, the following is specified.

When predFlagLX is equal to 1, the following applies.

* + The reference picture consisting of an ordered two-dimensional array refPicLXL of luma samples and two ordered two-dimensional arrays refPicLXCb and refPicLXCr of chroma samples is derived by invoking the process specified in subclause 8.5.2.2.1 with refIdxLX as input.
  + The arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr are derived by invoking the fractional sample interpolation process specified in subclause 8.5.2.2.2 with the luma locations ( xC, yC ), ( xB, yB ), the width and the height of the current luma prediction block nPbW, nPbH, the motion vectors mvLX, mvCLX, and the reference arrays with refPicLXL, refPicLXCb and refPicLXCr given as input.
  + When weighting\_factor\_index is unequal to 0 and refIdxLX points to a temporal reference picture, the inter-view residual prediction process as specified in subclause G.8.5.2.2.5 is invoked with the luma locations ( xC, yC ) and ( xB, yB ), the size of the current luma coding block nCS, the variables nPSW and nPSH, the reference view index refViewIdx set equal to 0, the prediction list utilization flag, predFlagLX, the reference index refIdxLX, luma motion vector mvLX, chroma motion vector mvCLX, the disparity vector mvDisp and the arrays predSamplesL, predSamplesCb, and predSamplesCr as the inputs and the outputs are modified versions of the arrays predSamplesL, predSamplesCb, and predSamplesCr.

…

~~When res\_pred\_flag is equal to 1, the inter-view residual prediction process as specified in subclause G.8.5.2.2.5 is invoked with the luma locations ( xC, yC ) and ( xB, yB ), the size of the current luma coding block nCS, the variables nPSW and nPSH, the reference view index refViewIdx set equal to 0, the prediction list utilization flags, predFlagL0 and predFlagL1, the reference indices refIdxL0 and refIdxL1, and the arrays predSamples~~~~L~~~~, predSamples~~~~Cb~~~~, and predSamples~~~~Cr~~ ~~as the inputs and the outputs are modified versions of the arrays predSamples~~~~L~~~~, predSamples~~~~Cb~~~~, and predSamples~~~~Cr~~~~.~~

G.8.5.2.2.5 Inter-view residual prediction process

The process is only invoked if weight\_factor\_index ~~res\_pred\_flag~~ is unequal to ~~1~~ 0.

Inputs to this process are:

* a luma location ( xC, yC ) specifying the top-left sample of the current luma coding block relative to the top left luma sample of the current picture,
* a luma location ( xP, yP ) of the top-left luma sample of the current prediction unit relative to the top-left luma sample of the current picture,
* a variable nCS specifying the size of the current luma coding block,
* variables nPSW and nPSH specifying the width and the height, respectively, of the current prediction unit,prediction list utilization flag~~s, predFlagL0~~predFlagLX ~~and predFlagL1~~,
* reference ~~indices~~index refIdxLX~~0 and refIdxL1~~, luma motion vector mvLX, chroma motion vector mvCLX and disparity vector mvDisp,
* a (nPSW)x(nPSH) array predSamplesL of luma prediction samples,
* two (nPSW / 2)x(nPSH / 2) arrays predSamplesCb and predSamplesCr of chroma prediction samples.

Output of this process are:

…

~~The derivation process for a disparity vector as specified in subclause G.8.5.2.1.13 is invoked with the luma locations ( xC, yC ) and ( xP, yP ), the coding block size nCS, the variables nPSW and nPSH, and the partition index partIdx as the inputs and the outputs are the view order index refViewIdx, the flag availableDV and the disparity vector mvDisp.[Ed. (GT) partIdx is missing as input to G.8.5.2.2.5].~~

~~[Ed. (GT). In software refViewIdx is set equal to the view order index of the view with minimal view\_id difference compared to the view\_id of the current view]. Let refResSamples~~~~L~~ ~~be the (PicWidthInSamples~~~~L~~~~)x(PicHeightInSamples~~~~L~~~~) array of constructed luma residual samples for inter-coded coding units for the view component with ViewIdx equal to refViewIdx. Let refResSamples~~~~Cb~~ ~~and refResSamples~~~~Cr~~ ~~be the (PicWidthInSamples~~~~L~~~~/ 2)x(PicHeightInSamples~~~~L~~~~/ 2) arrays of constructed Cb and Cr residual samples, respectively, for inter-coded coding units for the view component with ViewIdx equal to refViewIdx.~~

The following apply to derive the prediction values in the reference view:

* Set the reference picture sample arrays refPicLXL, refPicLXCb, and refPicLXCr corresponding to decoded sample arrays SL, SCb, SCr derived in subclause 8.7 for a previously-decoded picture which has the same POC value of current picture in the view component with ViewIdx equal to refViewIdx. The arrays refSamplesLXL, refSamplesLXCb, and refSamplesLXCr are derived by invoking the fractional sample interpolation process in subclause G.8.5.2.2.5.1 specified with the luma locations ( xC, yC ), ( xB, yB ), the width and the height of the current luma prediction block nPbW, nPbH, the motion vectors mvDisp, mvDisp, and the reference arrays with refPicLXL, refPicLXCb and refPicLXCr given as input.
* Set refPicLXL, refPicLXCb, and refPicLXCr corresponding to decoded sample arrays SL, SCb, SCr derived in subclause 8.7 for a previously-decoded picture which has the same POC value of RefPicListX[ refIdxLX ] in the view component with ViewIdx equal to refViewIdx. The arrays refPredSamplesLXL, refPredSamplesLXCb, and refPredSamplesLXCr are derived by invoking the fractional sample interpolation process specified in subclause G.8.5.2.2.5.1 with the luma locations ( xC, yC ), ( xB, yB ), the width and the height of the current luma prediction block nPbW, nPbH, the motion vectors (mvLX[0]+ mvDisp[0], mvLX[1]+ mvDisp[1]), (mvLX[0]+ mvDisp[0], mvLX[1]+ mvDisp[1]) and the reference arrays with refPicLXL, refPicLXCb and refPicLXCr given as input.

The modified prediction samples predSamplescom[ x ][ y ] with x = 0..(nPbW)−1 and y = 0..(nPbH)−1 are derived as follows:

predSamplescom[ x ][ y ] = predSamplescom[ x ][ y ] + ( refSamplesLXcom[ x ][ y ] - refPredSamplesLXcom[ x ][ y ] + offset ) >> shiftVal ) (x‑2)

where com could be replaced by L, Cb, or Cr. When com is unequal to L, nPbW is replaced by nPbW / 2 and nPbH is replaced by nPbH / 2, and the variable shiftVal and offset are defined as follows:

shiftVal  = (weight\_factor\_index == 2)? 1: 0 (x‑5)

offset = (refSamplesLXcom[ x ][ y ] - refPredSamplesLXcom[ x ][ y ]) > 0 ? 1: 0 (x‑6)

G.8.5.2.2.5.1 Fractional sample interpolation process

The same process as defined in HEVC subclause 8.5.3.2.2 is invoked with 8.5.3.2.2.1and 8.5.3.2.2.2 replaced by G.8.5.2.2.5.2, G.8.5.2.2.5.2, repsectively.

G.8.5.2.2.5.2 Bilinear interpolation for luma samples

Inputs to this process are:

– a luma/chroma location in full-sample units ( xIntCOM, yIntCOM ),

– a luma/chroma location offset in fractional-sample units ( xFracCOM, yFracCOM ),

– luma/chroma component samples from the selected reference picture refPicLXCOM.

Output of this process is a predicted chroma sample value predPartLXCOM[ xC, yC ].

Let the variable COM denotes the component of one picture with its value equal to L for luma and C for chroma component, respectively.

In Figure xx, the positions labelled with A, B, C, and D represent COM samples at full-sample locations inside the given two-dimensional array refPicLXCOM of chroma samples. When COM is equal to L, xFracCOM and yFracCOM are multiplied by 2.



Figure xx – Fractional sample position dependent variables in bi-linear interpolation and surrounding integer position samples A, B, C, and D

The sample coordinates specified in the following equations are used for generating the predicted COM sample value predPartLXCOM[ xCOM, yCOM ].

xAC = Clip3( 0, PicWidthInSamplesCOM − 1, xIntCOM ) (x)  
xBC = Clip3( 0, PicWidthInSamplesCOM − 1, xIntCOM + 1 ) (x)xCC = Clip3( 0, PicWidthInSamplesCOM − 1, xIntCOM ) (x)xDC = Clip3( 0, PicWidthInSamplesCOM − 1, xIntCOM + 1 ) (x)

yAC = Clip3( 0, refPicHeightEffectiveCOM − 1, yIntCOM ) (x)  
yBC = Clip3( 0, refPicHeightEffectiveCOM − 1, yIntCOM ) (x)yCC = Clip3( 0, refPicHeightEffectiveCOM − 1, yIntCOM + 1 ) (x)yDC = Clip3( 0, refPicHeightEffectiveCOM − 1, yIntCOM + 1 ) (x)

Given the COM samples A, B, C, and D at full-sample locations specified in above equations, the predicted COM sample value predPartLXCOM[ xC, yC ] is derived as:

predPartLXCOM[ xCOM, yCOM ] = ( ( 8 − xFracCOM ) \* ( 8 − yFracCOM ) \* A + xFracCOM \* ( 8 − yFracCOM ) \* B +  
 ( 8 − xFracC ) \* yFracCOM \* C + xFracCOM \* yFracCOM \* D ) >> 6 (x)

~~When the flag availableDV is equal to 0 the whole decoding process of this sub-clause terminates. The variable log2resPredDenom is set equal to 0 and the following ordered steps apply.~~

* 1. ~~When predFlagL0 is equal to 1 and ViewIdx is not equal to the view order index of RefPicListL0[ refIdxL0 ], log2resPredDenom is set equal to log2resPredDenom + 1.~~
  2. ~~When predFlagL1 is equal to 1 and ViewIdx is not equal to the view order index of RefPicListL1[ refIdxL1 ], log2resPredDenom is set equal to log2resPredDenom + 1.~~
  3. ~~The variable log2MaxResPredDenom is derived by~~
     1. ~~log2MaxResPredDenom = ( ( predFlagL0 = = 1 ) && ( predFlagL1 == 1 ) ? 1 : 0 ) (G‑188)~~

~~When log2resPredDenom is greater than log2MaxResPredDenom the whole decoding process of this sub-clause terminates.~~

~~For y proceeding over the values 0..(nPSH – 1) and x proceeding over the values 0..(nPSW – 1), the following ordered steps apply.~~

* 1. ~~The variables xR0, xR1, w0, and w1 are derived by~~
     + 1. ~~xR0 = Clip3( 0, PicWidthInSamples~~~~L~~~~– 1, xP + x + (mvDisp[ 0 ] >> 2 ) ) (G‑189)  
          xR1 = Clip3( 0, PicWidthInSamples~~~~L~~~~– 1, xP + x + (mvDisp[ 0 ] >> 2 ) + 1 ) (G‑190)  
          w0 = 4 – mvDisp[0] + ( ( mvDisp[0] >> 2 ) << 2 ) (G‑191)  
          w1 = mvDisp[0] − ( ( mvDisp[0] >> 2 ) << 2 ) (G‑192)~~
  2. ~~The sample predSamples~~~~L~~~~[ x, y ] is modified by~~
     + 1. ~~delta~~~~L~~ ~~= ( w0 \* refResSamples~~~~L~~~~[ xR0, y ] + w1 \* refResSamples~~~~L~~~~[ xR1, y ] + 4 ) >> 3 (G‑193)  
          predSamples~~~~L~~~~[ x, y ] = predSamples~~~~L~~~~[ x, y ] + ( delta~~~~L~~ ~~>> log2resPredDenom ) (G‑194)~~

~~For y proceeding over the values 0..(nPSH / 2 – 1) and x proceeding over the values 0..(nPSW / 2 – 1), the following ordered steps are specified:~~

* 1. ~~The variables xR0, xR1, w0, and w1 are derived by~~
     + 1. ~~xR0 = Clip3( 0, PicWidthInSamples~~~~L~~~~/ 2 – 1, xP / 2 + x + (mvDisp[0] >> 3 ) ) (G‑195)  
          xR1 = Clip3( 0, PicWidthInSamples~~~~L~~~~/ 2 – 1, xP / 2 + x + (mvDisp[0] >> 3 ) + 1 ) (G‑196)  
          w0 = 8 – mvDisp[0] + ( (mvDisp[0] >> 3 ) << 3 ) (G‑197)  
          w1 = mvDisp[0] − ( (mvDisp[0] >> 3 ) << 3 ) (G‑198)~~
  2. ~~The sample predSamples~~~~Cb~~~~[ x, y ] is modified by~~
     + 1. ~~delta~~~~Cb~~ ~~= ( w0 \* refResSamples~~~~Cb~~~~[ xR0, y ] + w1 \* refResSamples~~~~Cb~~~~[ xR1, y ] + 8 ) >> 4 (G‑199)  
          predSamples~~~~Cb~~~~[ x, y ] = predSamples~~~~Cb~~~~[ x, y ] + ( delta~~~~Cb~~  ~~>> log2resPredDenom ) (G‑200)~~
  3. ~~The sample predSamples~~~~Cr~~~~[ x, y ] is modified by~~
     + 1. ~~delta~~~~Cr~~ ~~= ( w0 \* refResSamples~~~~Cr~~~~[ xR0, y ] + w1 \* refResSamples~~~~Cr~~~~[ xR1, y ] + 8 ) >> 4 (G‑201)  
          predSamples~~~~Cr~~~~[ x, y ] = predSamples~~~~Cr~~~~[ x, y ] + ( delta~~~~Cr~~  ~~>> log2resPredDenom ) (G‑202)~~

G.9.3.1.1 Initialization process for context variables

Table G‑6 – Association of ctxIdx and syntax elements for each initializationType in the initialization process

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| coding\_unit() | **Syntax element** | **ctxIdxTable** | **initType** | | |
| **0** | **1** | **2** |
| dmm\_dc\_1\_abs dmm\_dc\_2\_abs | Table G‑11 | 0 | 1 | 2 |
| weighting\_factor\_index ~~res\_pred\_flag~~ | Table G‑12 | ~~0~~ | ~~1~~0..2 | ~~2~~3..5 |
| … | … |  |  |  |

Table G‑12 – Values of variable initValue for weighting\_factor\_index ~~res\_pred\_flag~~ ctxIdx

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Initialization variable** | weighting\_factor\_index | | | | | |
| **0** | **1** | **2** | **3** | **4** | **5** |
| **initValue** | 154 | 154 | 154 | 154 | 154 | 154 |

G.9.3.2 Binarization process

Table G‑20 – Syntax elements and associated types of binarization, maxBinIdxCtx, ctxIdxTable, and ctxIdxOffset

| **Syntax element** | **initType** | **Type of binarization** | **maxBinIdxCtx** | **ctxIdxTable** | **ctxIdxOffset** |
| --- | --- | --- | --- | --- | --- |
| **…** |  |  |  |  |  |
| ~~res\_pred\_flag~~ weighting\_factor\_index | 1 | ~~FL~~TU, cMax = ~~1~~2 | 0 | Table G‑12 | 1 |
| 2 | 0 | Table G‑12 | 2 |

G.9.2.3.1.1 Derivation process of ctxIdxInc using left and above syntax elements

Table G‑22 – Specification of ctxIdxInc using left and above syntax elements

|  |  |  |  |
| --- | --- | --- | --- |
| **Syntax element** | **condL** | **condA** | **ctxIdxInc** |
| sdc\_flag | sdc\_flag [ xL ][ yL ] | sdc\_flag[ xA ][ yA ] | ( condL && availableL ) + ( condA && availableA ) |
| weighting\_factor\_index | weighting\_factor\_index [ xL ][ yL ] | weighting\_factor\_index [ xA ][ yA ] | ( condL && availableL ) + ( condA && availableA ) |