|  |  |
| --- | --- |
| **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**  2nd Meeting: Shanghai, CN, 13–19 Oct. 2012 | Document: JCT3V-B0054 |

|  |  |
| --- | --- |
| *Title:* | **3D-CE1.a: Generalized view synthesis prediction (GVSP) mode** |
| *Status:* | Input Document |
| *Purpose:* | Proposal |
| *Source:* | Qualcomm Incorporated |

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

# Abstract

As an attachment of JCT3V-B0054, this document provides detailed syntax design for the proposed method.

# Semantics

In the following sub-sections, the flag vsp\_flag is defined for an MB partition of any size.

### 7.3.5 Macroblock layer syntax

*In 7.3.5,Macroblock layer syntax, replace the syntax table with:*

|  |  |  |
| --- | --- | --- |
| macroblock\_layer( ) { | **C** | **Descriptor** |
| if (slice\_vsp\_flag && nal\_unit\_type = = 21 && !DepthFlag && VspRefExist) |  |  |
| **vsp\_flag** | 2 | ae(v) |
| if (!vsp\_ flag){ |  |  |
| if (!mb\_ivmp\_flag) { |  |  |
| if( DepthFlag && disp\_flag ) |  |  |
| **mb\_disp\_flag** |  | u(1) | ae(v) |
| if(!mb\_disp\_flag){ |  |  |
| **mb\_type** | 2 | ue(v) | ae(v) |
| if(nal\_unit\_type = = 21 && !DepthFlag   && slice\_type = = B   && direct\_spatial\_mv\_pred\_flag && VspRefExist  && mb\_type = = B\_Direct\_16x16 ) |  |  |
| **mb\_direct\_type\_flag** | 2 | u(1) | ae(v) |
| } |  |  |
| } |  |  |
| } |  |  |
| if( !mb\_disp\_flag ) { |  |  |
| if( mb\_type = = I\_PCM ) { |  |  |
| while( !byte\_aligned( ) ) |  |  |
| **pcm\_alignment\_zero\_bit** | 3 | f(1) |
| for( i = 0; i < 256; i++ ) |  |  |
| **pcm\_sample\_luma[** i **]** | 3 | u(v) |
| for( i = 0; i < 2 \* MbWidthC \* MbHeightC; i++ ) |  |  |
| **pcm\_sample\_chroma[** i **]** | 3 | u(v) |
| } else { |  |  |
| noSubMbPartSizeLessThan8x8Flag = 1 |  |  |
| if( mb\_type != I\_NxN &&  MbPartPredMode( mb\_type, 0 ) != Intra\_16x16 &&  NumMbPart( mb\_type ) = = 4 ) { |  |  |
| if( !mb\_ivmp\_flag && !vsp\_flag ) |  |  |
| sub\_mb\_pred( mb\_type ) | 2 |  |
| for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ ) |  |  |
| if( sub\_mb\_type[ mbPartIdx ] != B\_Direct\_8x8 ) { |  |  |
| if( NumSubMbPart( sub\_mb\_type[ mbPartIdx ] ) > 1 ) |  |  |
| noSubMbPartSizeLessThan8x8Flag = 0 |  |  |
| } else if( !direct\_8x8\_inference\_flag ) |  |  |
| noSubMbPartSizeLessThan8x8Flag = 0 |  |  |
| } else { |  |  |
| if( transform\_8x8\_mode\_flag && mb\_type = = I\_NxN ) |  |  |
| **transform\_size\_8x8\_flag** | 2 | u(1) | ae(v) |
| if( !mb\_ivmp\_flag && !vsp\_flag ) |  |  |
| mb\_pred( mb\_type ) | 2 |  |
| } |  |  |
| if( MbPartPredMode( mb\_type, 0 ) != Intra\_16x16 ) { |  |  |
| **coded\_block\_pattern** | 2 | me(v) | ae(v) |
| if( CodedBlockPatternLuma > 0 &&  transform\_8x8\_mode\_flag && mb\_type != I\_NxN &&  noSubMbPartSizeLessThan8x8Flag &&  (vsp\_flag || (!vsp\_flag && (mb\_type != B\_Direct\_16x16 | | direct\_8x8\_inference\_flag ) ) ) ) |  |  |
| **transform\_size\_8x8\_flag** | 2 | u(1) | ae(v) |
| } |  |  |
| if( CodedBlockPatternLuma > 0 | |   CodedBlockPatternChroma > 0 | |  MbPartPredMode( mb\_type, 0 ) = = Intra\_16x16 ) { |  |  |
| **mb\_qp\_delta** | 2 | se(v) | ae(v) |
| residual( 0, 15 ) | 3 | 4 |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**vsp\_flag** equal to 1 indicates that the whole MB is predicted from a view synthesized picture. This flag equal to 0 indicates that the whole MB may be predicted by other modes. When this flag is equal to 1, the mb\_type is not signaled (the whole whole MB block is predicted from view synthesized picture). When not present, it is inferred to be equal to 0.

#### 7.3.5.1 Macroblock prediction syntax

|  |  |  |
| --- | --- | --- |
| mb\_pred( mb\_type ) { | **C** | **Descriptor** |
| if( MbPartPredMode( mb\_type, 0 ) = = Intra\_4x4 | |   MbPartPredMode( mb\_type, 0 ) = = Intra\_8x8 | |   MbPartPredMode( mb\_type, 0 ) = = Intra\_16x16 ) { |  |  |
| if( MbPartPredMode( mb\_type, 0 ) = = Intra\_4x4 ){ |  |  |
| for( luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++ ) { |  |  |
| **prev\_intra4x4\_pred\_mode\_flag[** luma4x4BlkIdx **]** | 2 | u(1) | ae(v) |
| if( !prev\_intra4x4\_pred\_mode\_flag**[** luma4x4BlkIdx **]** ) |  |  |
| **rem\_intra4x4\_pred\_mode[** luma4x4BlkIdx **]** | 2 | u(3) | ae(v) |
| } |  |  |
| for( luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++ ) { |  |  |
| if( psip\_flag && Psip4x4Available  && Intra4x4PredMode[luma4x4BlkIdx] = = 2 ){ |  |  |
| **psip4x4\_dir**[ luma4x4BlkIdx ] | 2 | u(1) |
| for( idxRowCol=0; idxRowCol<4; idxRowCol++ ) |  |  |
| **psip4x4\_code\_diff**[ luma4x4BlkIdx ][ idxRowCol ] | 2 | se(v) | ae(v) |
| [Ed. CAVLC syntax missing] |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| if( MbPartPredMode( mb\_type, 0 ) = = Intra\_8x8 ){ |  |  |
| for( luma8x8BlkIdx=0; luma8x8BlkIdx<4; luma8x8BlkIdx++ ) { |  |  |
| **prev\_intra8x8\_pred\_mode\_flag[** luma8x8BlkIdx **]** | 2 | u(1) | ae(v) |
| if( !prev\_intra8x8\_pred\_mode\_flag[ luma8x8BlkIdx ] ) |  |  |
| **rem\_intra8x8\_pred\_mode[** luma8x8BlkIdx **]** | 2 | u(3) | ae(v) |
| } |  |  |
| for( luma8x8BlkIdx =0; luma8x8BlkIdx <4; luma8x8BlkIdx ++ ) { |  |  |
| if( psip\_flag && Psip8x8Available   && Intra8x8PredMode[luma4x4BlkIdx] = = 2 ){ |  |  |
| **psip8x8\_dir**[ luma8x8BlkIdx ] | 2 | u(1) |
| for( idxRowCol=0; idxRowCol<8; idxRowCol++ ) |  |  |
| **psip8x8\_code\_diff**[ luma8x8BlkIdx ][idxRowCol] | 2 | se(v) | ae(v) |
| } |  |  |
| } |  |  |
| } |  |  |
| if( MbPartPredMode( mb\_type, 0 ) = = Intra\_16x16 ) { |  |  |
| if( psip\_flag && Psip16x16Available   && Intra16x16PredMode = = 2 ){ |  |  |
| **psip16x16\_dir** | 2 | u(1) |
| for( idxRowCol=0; idxRowCol<16; idxRowCol++ ) |  |  |
| **psip16x16\_code\_diff**[ idxRowCol ] | 2 | se(v) | ae(v) |
| } |  |  |
| } |  |  |
| if( ChromaArrayType = = 1 | | ChromaArrayType = = 2 ) |  |  |
| **intra\_chroma\_pred\_mode** | 2 | ue(v) | ae(v) |
| } else if( MbPartPredMode( mb\_type, 0 ) != Direct ) { |  |  |
| for( mbPartIdx = 0; mbPartIdx < NumMbPart( mb\_type ); mbPartIdx++) |  |  |
| if (slice\_vsp\_flag && NumMbPart( mb\_type ) != 1&& nal\_unit\_type = = 21 && !DepthFlag && VspRefExist) { |  |  |
| **vsp\_flag[** mbPartIdx **]** | 2 | u(1) | ae(v) |
| } |  |  |
| if( !vsp\_flag[ mbPartIdx ] && ( num\_ref\_idx\_l0\_active\_minus1 > 0 | | mb\_field\_decoding\_flag != field\_pic\_flag ) &&   MbPartPredMode( mb\_type, mbPartIdx ) != Pred\_L1 ) |  |  |
| **ref\_idx\_l0[** mbPartIdx **]** | 2 | te(v) | ae(v) |
| for(mbPartIdx = 0; mbPartIdx < NumMbPart( mb\_type ); mbPartIdx++) |  |  |
| if( !vsp\_flag[ mbPartIdx ] && ( num\_ref\_idx\_l1\_active\_minus1 > 0 | | mb\_field\_decoding\_flag != field\_pic\_flag ) &&   MbPartPredMode( mb\_type, mbPartIdx ) != Pred\_L0 ) |  |  |
| **ref\_idx\_l1[** mbPartIdx **]** | 2 | te(v) | ae(v) |
| for( mbPartIdx = 0; mbPartIdx < NumMbPart( mb\_type ); mbPartIdx++) |  |  |
| if( !vsp\_flag[ mbPartIdx ] && MbPartPredMode ( mb\_type, mbPartIdx ) != Pred\_L1 ) |  |  |
| for( compIdx = 0; compIdx < 2; compIdx++ ) |  |  |
| **mvd\_l0[** mbPartIdx **][** 0 **][** compIdx **]** | 2 | se(v) | ae(v) |
| for( mbPartIdx = 0; mbPartIdx < NumMbPart( mb\_type ); mbPartIdx++) |  |  |
| if( !vsp\_flag[ mbPartIdx ] && MbPartPredMode( mb\_type, mbPartIdx ) != Pred\_L0 ) |  |  |
| for( compIdx = 0; compIdx < 2; compIdx++ ) |  |  |
| **mvd\_l1[** mbPartIdx **][** 0 **][** compIdx **]** | 2 | se(v) | ae(v) |
| } |  |  |
| } |  |  |

**vsp\_flag[** mbPartIdx **]** equal to 1 indicates that the current MB partition is predicted from a view synthesized picture. This flag equal to 0 indicates that the whole MB partition is not predicted from a view synthesized picture. When not present, this flag is inferred to be equal to 0.

#### 7.3.5.2 Sub-macroblock prediction syntax

|  |  |  |
| --- | --- | --- |
| sub\_mb\_pred( mb\_type ) { | **C** | **Descriptor** |
| for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ ) { |  |  |
| if (slice\_vsp\_flag && nal\_unit\_type = = 21 && !DepthFlag  && VspRefExist) // vsp\_mb\_flag is not 1 |  |  |
| **vsp\_flag[** mbPartIdx **]** |  |  |
| if (!vsp\_flag**[** mbPartIdx **]**) |  |  |
| **sub\_mb\_type[** mbPartIdx **]** | 2 | ue(v) | ae(v) |
| } |  |  |
| for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ ) |  |  |
| if( !**vsp\_flag**[ mbPartIdx ]) && ( num\_ref\_idx\_l0\_active\_minus1 > 0 | | mb\_field\_decoding\_flag ) &&  mb\_type != P\_8x8ref0 &&  sub\_mb\_type[ mbPartIdx ] != B\_Direct\_8x8 &&  SubMbPredMode( sub\_mb\_type[ mbPartIdx ] ) != Pred\_L1 ) |  |  |
| **ref\_idx\_l0[** mbPartIdx **]** | 2 | te(v) | ae(v) |
| for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ ) |  |  |
| if( ! **vsp\_flag**[ mbPartIdx ]) && (num\_ref\_idx\_l1\_active\_minus1 > 0 | | mb\_field\_decoding\_flag ) &&  sub\_mb\_type[ mbPartIdx ] != B\_Direct\_8x8 &&  SubMbPredMode( sub\_mb\_type[ mbPartIdx ] ) != Pred\_L0 ) |  |  |
| **ref\_idx\_l1[** mbPartIdx **]** | 2 | te(v) | ae(v) |
| for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ ) |  |  |
| if( !**vsp\_flag**[ mbPartIdx ]) &&( sub\_mb\_type[ mbPartIdx ] != B\_Direct\_8x8 &&  SubMbPredMode( sub\_mb\_type[ mbPartIdx ] ) != Pred\_L1 ) |  |  |
| for( subMbPartIdx = 0;   subMbPartIdx < NumSubMbPart( sub\_mb\_type[ mbPartIdx ] );  subMbPartIdx++) |  |  |
| for( compIdx = 0; compIdx < 2; compIdx++ ) |  |  |
| **mvd\_l0[** mbPartIdx **][** subMbPartIdx **][** compIdx **]** | 2 | se(v) | ae(v) |
| for( mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++ ) |  |  |
| if( !**vsp\_flag**[ mbPartIdx ]) &&sub\_mb\_type[ mbPartIdx ] != B\_Direct\_8x8 &&  SubMbPredMode( sub\_mb\_type[ mbPartIdx ] ) != Pred\_L0 ) |  |  |
| for( subMbPartIdx = 0;   subMbPartIdx < NumSubMbPart( sub\_mb\_type[ mbPartIdx ] );  subMbPartIdx++) |  |  |
| for( compIdx = 0; compIdx < 2; compIdx++ ) |  |  |
| **mvd\_l1[** mbPartIdx **][** subMbPartIdx **][** compIdx **]** | 2 | se(v) | ae(v) |
| } |  |  |

8.3.1.3 Derivation process for luma motion vectors for VSP predicted macroblocks in P and SP slices

Input to this process is current macroblock partition index mbPartIdx.

This process is invoked when vsp\_flag[ mbPartIdx ] is equal to 1, nal\_unit\_type is equal to 21 and DepthFlag is equal to 0.

Outputs of this process are the motion vector mvL0 and the reference index refIdxL0.

The reference index refIdxL0 for a VSP predicted macroblock is derived as the synthetic picture that appears first in RefPicList0.

The motion vector mvL0 is set equal to zero motion vector.

8.3.1.4 Derivation process for luma motion vectors for VSP predicted macroblocks in B slices

Input to this process is current macroblock partition index mbPartIdx.

This process is invoked when vsp\_flag[ mbPartIdx ] is equal to 1, nal\_unit\_type is equal to 21 and DepthFlag is equal to 0.

Outputs of this process are the reference indices refIdxL0, refIdxL1, the motion vectors mvL0 and mvL1.

The reference index refIdxL0 for a VSP predicted macroblock partition mbPartIdx is derived as the synthetic picture that appears first in RefPicList0. The reference index refIdxL1 for a VSP predicted macroblock partition mbPartIdx is derived as the synthetic picture that appears first in RefPicList1.

The motion vectors mvL0 and mvL1 are set equal to zero motion vectors.