**Draft Text Specification**

The proposed text changes are based on the document JCTVC-P1005-v1.doc for the Intra BC prediction in SCCE1 Test 1.1 with Intra BC search area is 2 CTUs for inter slices and full frame for intra slices. The changes are marked in yellow.

**Syntax**

**Decoding Process**

**8.4.4 Derivation process for block vector components in intra block copying prediction mode**

Inputs to this process are:

* a luma location ( xCb, yCb ) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
* a variable log2CbSize specifying the size of the current luma coding block.

Output of this process is the (nCbS)x(nCbX) array of block vectors bvIntra.

The variables nCbS, nCbSw, nCbSh are derived as follows:

nCbS = 1  <<  log2CbSize (8‑25)

nPbSw = nCbS / ( PartMode = = PART\_2Nx2N | | PartMode = = PART\_2NxN ? 1 : 2 ) (8‑25)

nPbSh = nCbS / ( PartMode = = PART\_2Nx2N | | PartMode = = PART\_Nx2N ? 1 : 2 ) (8‑25)

The variable BvpIntra[ compIdx ] specifies a block vector predictor. The horizontal block vector component is assigned compIdx = 0 and the vertical block vector component is assigned compIdx = 1.

Depending upon PartMode, the variable numPartitions is derived as follows:

– If PartMode is equal to PART\_2Nx2N, numPartitions is set equal to 1.

– Otherwise, if PartMode is equal to either PART\_2NxN or PART\_Nx2N, numPartitions is set equal to 2.

– Otherwise (PartMode is equal to PART\_NxN), numPartitions is set equal to 4.

The array of block vectors bvIntra is derived by the following ordered steps, for the variable blkIdx proceeding over the values 0..( numPartitions − 1 ):

1. The variable blkInc is set equal to ( PartMode = = PART\_2NxN ? 2 : 1 ).
2. The variable xPb is set equal to xCb + nPbSw \* ( blkIdx \* blkInc % 2 ).
3. The variable yPb is set equal to yCb + nPbSh \* ( blkIdx / 2 )
4. The following ordered steps apply, for the variable compIdx proceeding over the values 0..1:
5. Depending upon the number of times this process has been invoked for the current coding tree unit, the following applies:

* If this process is invoked for the first time for the current coding tree unit, bvIntra[ xPb ][ yPb ][ compIdx ] is derived as follows:

bvIntra[ xPb ][ yPb ][ 0 ] = BvdIntra[ xPb ][ yPb ][ 0 ] − nCbS (8‑25)

bvIntra[ xPb ][ yPb ][ 1 ] = BvdIntra[ xPb ][ yPb ][ 1 ] (8‑25)

* Otherwise, bvIntra[ xPb ][ yPb ][ compIdx ] is derived as follows:

bvIntra[ xPb ][ yPb ][ 0 ] = BvdIntra[ xPb ][ yPb ][ 0 ] + BvpIntra[ 0 ] (8‑25)

bvIntra[ xPb ][ yPb ][ 1 ] = BvdIntra[ xPb ][ yPb ][ 1 ] + BvpIntra[ 1 ] (8‑25)

1. The value of BvpIntra[ compIdx ] is updated to be equal to bvIntra[ xPb ][ yPb ][ compIdx ].
2. For use in derivation processes of variables invoked later in the decoding process, the following assignments are made for x = 0..nPbSw − 1 and y = 0..nPbSh − 1:

bvIntra[ xPb + x ][ yPb + y ][ compIdx ] = bvIntra[ xPb ][ yPb ][ compIdx ] (8‑25)

For P and B slice types, it is a requirement of bitstream conformance that all of the the following conditions are true:

– The value of bvIntra[ xPb ][ yPb ][ 0 ] shall be greater than or equal to – ( xPb % CtbSizeY + 64 ).

– The value of bvIntra[ xPb ][ yPb ][ 1 ] shall be greater than or equal to – ( yPb % CtbSizeY).

– When the derivation process for z-scan order block availability as specified in subclause 6.4.1 is invoked with ( xCurr, yCurr ) set equal to ( xCb, yCb ) and the neighbouring luma location ( xNbY, yNbY ) set equal to ( xPb + bvIntra[ xPb ][ yPb ][ 0 ], yPb + bvIntra[ xPb ][ yPb ][ 1 ] ) as inputs, the output shall be equal to TRUE.

– When the derivation process for z-scan order block availability as specified in subclause 6.4.1 is invoked with ( xCurr, yCurr ) set equal to ( xCb, yCb ) and the neighbouring luma location ( xNbY, yNbY ) set equal to ( xPb + bvIntra[ xPb ][ yPb ][ 0 ] + nPbSw − 1, yPb + bvIntra[ xPb ][ yPb ][ 1 ] + nPbSh – 1 ) as inputs, the output shall be equal to TRUE.

– When the slice type is I and entropy\_coding\_sync\_enabled\_flag is equal to 1, the following condition shall be true

– The value of xPb % CtbSizeY + bvIntra[ xPb ][ yPb ][ 0 ] shall be less than or equal to (CurrCtby – RefCtby) << 1

where CurrCtby =  (yPb/ CtbSizeY) \* CtbSizeY

RefCtby = ((yPb + bvIntra[ xPb ][ yPb ][ 1 ])/ CtbSizeY) \* CtbSizeY

– One or both of the following conditions shall be true:

– bvIntra[ xPb ][ yPb ][ 0 ] + nPbSw <= 0

– bvIntra[ xPb ][ yPb ][ 1 ] + nPbSh <= 0