## Decoding process for coding units coded in inter prediction mode

### General decoding process for coding units coded in inter prediction mode

Inputs to this process are:

* a luma location ( xCb, yCb ) specifying the top-left sample of the current coding block relative to the top‑left luma sample of the current picture,
* a variable cbWidth specifying the width of the current coding block in luma samples,
* a variable cbHeight specifying the height of the current coding block in luma samples,
* a variable treeType specifying whether a single or a dual tree is used and if a dual tree is used, it specifies whether the current tree corresponds to the luma or chroma components.

Output of this process is a modified reconstructed picture before in-loop filtering.

The variable currPic specifies the current picture.

The derivation process for quantization parameters as specified in subclause 8.7.1 is invoked with the luma location ( xCb, yCb ), the width of the current coding block in luma samples cbWidth and the height of the current coding block in luma samples cbHeight, and the variable treeType as inputs.

The decoding process for coding units coded in inter prediction mode consists of the following ordered steps:

1. The variable dmvrFlag is set equal to 0, the variables cbProfFlagL0 and cbProfFlagL1 are both set equal to 0, and the variable hpelIfIdx is set equal to 0.
2. The motion vector components and reference indices of the current coding unit are derived as follows:

* If MergeGpmFlag[ xCb ][ yCb ], inter\_affine\_flag[ xCb ][ yCb ] and merge\_subblock\_flag[ xCb ][ yCb ] are all equal to 0, the following applies:
* The derivation process for motion vector components and reference indices as specified in subclause 8.5.2.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight as inputs, and the luma motion vectors mvL0[ 0 ][ 0 ] and mvL1[ 0 ][ 0 ], the reference indices refIdxL0 and refIdxL1 and the prediction list utilization flags predFlagL0[ 0 ][ 0 ] and predFlagL1[ 0 ][ 0 ], the half sample interpolation filter index hpelIfIdx, and the bi-prediction weight index bcwIdx as outputs.
* When all of the following conditions are true, dmvrFlag is set equal to 1:
* ph\_dmvr\_disabled\_flag is equal to 0.
* general\_merge\_flag[ xCb ][ yCb ] is equal to 1.
* both predFlagL0[ 0 ][ 0 ] and predFlagL1[ 0 ][ 0 ] are equal to 1.
* mmvd\_merge\_flag[ xCb ][ yCb ] is equal to 0.
* ciip\_flag[ xCb ][ yCb ] is equal to 0.
* DiffPicOrderCnt( currPic, RefPicList[ 0 ][ refIdxL0 ]) is equal to DiffPicOrderCnt( RefPicList[ 1 ][ refIdxL1 ], currPic ).
* RefPicList[ 0 ][ refIdxL0 ] is an STRP and RefPicList[ 1 ][ refIdxL1 ] is an STRP.
* bcwIdx is equal to 0.
* Both luma\_weight\_l0\_flag[ refIdxL0 ] and luma\_weight\_l1\_flag[ refIdxL1 ] are equal to 0.
* Both chroma\_weight\_l0\_flag[ refIdxL0 ] and chroma\_weight\_l1\_flag[ refIdxL1 ] are equal to 0.
* cbWidth is greater than or equal to 8.
* cbHeight is greater than or equal to 8.
* cbHeight\*cbWidth is greater than or equal to 128.
* RprConstraintsActiveFlag[ 0 ][ refIdxL0 ] is equal to 0 and RprConstraintsActiveFlag[ 1 ][ refIdxL1 ] is equal to 0.
* If dmvrFlag is equal to 1, the following applies:
* For X = 0..1, 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 8.5.6.2 with X and refIdxLX as inputs.
* The number of luma subblocks in horizontal direction numSbX and in vertical direction numSbY, the subblock width sbWidth and the subblock height sbHeight are derived as follows:

numSbX = ( cbWidth > 16 ) ? ( cbWidth  >>  4 ) : 1 (442)

numSbY = ( cbHeight > 16 ) ? ( cbHeight  >>  4 ) : 1 (443)

sbWidth = ( cbWidth > 16 ) ? 16 : cbWidth (444)

sbHeight = ( cbHeight > 16 ) ? 16 : cbHeight (445)

* For xSbIdx = 0..numSbX − 1 and ySbIdx = 0..numSbY − 1, the following applies:
* The luma motion vectors mvLX[ xSbIdx ][ ySbIdx ] and the prediction list utilization flags predFlagLX[ xSbIdx ][ ySbIdx ] with X = 0..1, and the luma location ( xSb[xSbIdx][ySbIdx], ySb[xSbIdx][ySbIdx] ) specifying the top-left sample of the subblock relative to the top‑left luma sample of the current picture are derived as follows:

mvLX[ xSbIdx ][ ySbIdx ] = mvLX[ 0 ][ 0 ] (446)

predFlagLX[ xSbIdx ][ ySbIdx ] = predFlagLX[ 0 ][ 0 ] (447)

xSb[ xSbIdx ][ ySbIdx ] =  xCb + xSbIdx \* sbWidth (448)

ySb[ xSbIdx ][ ySbIdx ] =  yCb + ySbIdx \* sbHeight (449)

* The decoder-side motion vector refinement process specified in subclause 8.5.3.1 is invoked with xSb[ xSbIdx ][ ySbIdx ], ySb[ xSbIdx ][ ySbIdx ], sbWidth, sbHeight, the motion vectors mvLX[ xSbIdx ][ ySbIdx ] and the reference picture array refPicLXL as inputs and delta motion vectors dMvLX[ xSbIdx ][ ySbIdx ] with X = 0..1, and the mimimum sum of absolute difference in decoder-side motion vector refinement process dmvrSad[ xSbIdx ][ ySbIdx ] as outputs.
* When sps\_chroma\_format\_idc is not equal to 0, the derivation process for chroma motion vectors in subclause 8.5.2.13 is invoked with mvLX[ xSbIdx ][ ySbIdx ] with X = 0..1 as inputs, and mvCLX[ xSbIdx ][ ySbIdx ] with X = 0..1 as outputs.
* Otherwise (dmvrFlag is equal to 0), the following applies:
* For X = 0..1, when sps\_chroma\_format\_idc is not equal to 0, and treeType is equal to SINGLE\_TREE, and predFlagLX[ 0 ][0 ] is equal to 1, the derivation process for chroma motion vectors in subclause 8.5.2.13 is invoked with mvLX[ 0 ][ 0 ] as input, and mvCLX[ 0 ][ 0 ] as output.
* The number of luma subblocks in horizontal direction numSbX and in vertical direction numSbY are both set equal to 1.
* Otherwise, if MergeGpmFlag[ xCb ][ yCb ] is equal to 1, inter\_affine\_flag[ xCb ][ yCb ] and merge\_subblock\_flag[ xCb ][ yCb ] are both equal to 0, the following applies:
* The derivation process for geometric partitioning mode motion vector components and reference indices as specified in subclause 8.5.4.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight as inputs, and the luma motion vectors mvA and mvB, the chroma motion vectors mvCA and mvCB, the reference indices refIdxA and refIdxB and the prediction list flags predListFlagA and predListFlagB as outputs.
* The number of luma subblocks in horizontal direction numSbX and in vertical direction numSbY are both set equal to 1.
* Otherwise (inter\_affine\_flag[ xCb ][ yCb ] or merge\_subblock\_flag[ xCb ][ yCb ] is equal to 1), the derivation process for subblock motion vector components and reference indices as specified in subclause 8.5.5.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth, the luma coding block height cbHeight as inputs, and the reference indices refIdxL0 and refIdxL1, the number of luma subblocks in horizontal direction numSbX and in vertical direction numSbY, the prediction list utilization flags predFlagLX[ xSbIdx ][ ySbIdx ], the luma motion vector array mvLX[ xSbIdx ][ ySbIdx ], and the chroma motion vector array mvCLX[ xSbIdx ][ ySbIdx ] with xSbIdx = 0..numSbX − 1, and ySbIdx = 0..numSbY − 1, and with X = 0..1, the bi-prediction weight index bcwIdx, the prediction refinement utility flags cbProfFlagL0 and cbProfFlagL1, and motion vector difference arrays diffMvL0[ xIdx ][ yIdx ] and diffMvL1[ xIdx ][ yIdx ] with xIdx = 0.. cbWidth/numSbX − 1, yIdx = 0.. cbHeight/numSbY − 1 as outputs.

1. The arrays of luma and chroma motion vectors after decoder-side motion vector refinement, refMvLX[ xSbIdx ][ ySbIdx ] and refMvCLX[ xSbIdx ][ ySbIdx ], with X = 0..1, are derived as follows for xSbIdx = 0..numSbX − 1, ySbIdx = 0..numSbY − 1:

* If dmvrFlag is equal to 1, the derivation process for chroma motion vectors in subclause 8.5.2.13 is invoked with refMvLX[ xSbIdx ][ ySbIdx ] as inputs, and refMvCLX[ xSbIdx ][ ySbIdx ] as output and the input refMvLX[ xSbIdx ][ ySbIdx ] is derived as follows;

refMvLX[ xSbIdx ][ ySbIdx ] = mvLX[ xSbIdx ][ ySbIdx ] + dMvLX[ xSbIdx ][ ySbIdx ] (450)

refMvLX[ xSbIdx ][ ySbIdx ][ 0 ] = Clip3( −217, 217 − 1, refMvLX[ xSbIdx ][ ySbIdx ][ 0 ] ) (451)

refMvLX[ xSbIdx ][ ySbIdx ][ 1 ] = Clip3( −217, 217− 1, refMvLX[ xSbIdx ][ ySbIdx ][ 1 ] ) (452)

* Otherwise (dmvrFlag is equal to 0), the following applies:

refMvLX[ xSbIdx ][ ySbIdx ] = mvLX[ xSbIdx ][ ySbIdx ] (453)

refMvCLX [ xSbIdx ][ ySbIdx ] = mvCLX[ xSbIdx ][ ySbIdx ] (454)

NOTE – The array refMvLX is stored in MvDmvrLX and used in the derivation process for collocated motion vectors in subclause 8.5.2.12. The array of non-refine luma motion vectors MvLX is used in the spatial motion vector prediction and deblocking boundary filtering strength derivation processes.

1. The prediction samples of the current coding unit are derived as follows:

* If MergeGpmFlag[ xCb ][ yCb ] is equal to 0, the prediction samples of the current coding unit are derived as follows:
  + - * The decoding process for inter blocks as specified in subclause 8.5.6.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight, the number of luma subblocks in horizontal direction numSbX and in vertical direction numSbY, the luma motion vectors mvL0[ xSbIdx ][ ySbIdx ] and mvL1[ xSbIdx ][ ySbIdx ], and the refined luma motion vectors refMvL0[ xSbIdx ][ ySbIdx ] and refMvL1[ xSbIdx ][ ySbIdx ] with xSbIdx = 0..numSbX − 1, and ySbIdx = 0..numSbY − 1, the reference indices refIdxL0 and refIdxL1, the prediction list utilization flags predFlagL0[ xSbIdx ][ ySbIdx ] and predFlagL1[ xSbIdx ][ ySbIdx ], the half sample interpolation filter index hpelIfIdx, the bi-prediction weight index bcwIdx, the mimimum sum of absolute difference values in decoder-side motion vector refinement process dmvrSad[ xSbIdx ][ ySbIdx ], the decoder-side motion vector refinement flag dmvrFlag, the variable cIdx set equal to 0, the prediction refinement utility flags cbProfFlagL0 and cbProfFlagL1, and motion vector difference arrays diffMvL0[ xIdx ][ yIdx ] and diffMvL1[ xIdx ][ yIdx ] with xIdx = 0..cbWidth / numSbX − 1, and yIdx = 0..cbHeight / numSbY − 1 as inputs, and the inter prediction samples (predSamples) that are an (cbWidth)x(cbHeight) array predSamplesL of prediction luma samples as outputs.
      * When sps\_chroma\_format\_idc is not equal to 0. the decoding process for inter blocks as specified in subclause 8.5.6.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight, the number of luma subblocks in horizontal direction numSbX and in vertical direction numSbY, the chroma motion vectors mvCL0[ xSbIdx ][ ySbIdx ] and mvCL1[ xSbIdx ][ ySbIdx ], and the refined chroma motion vectors refMvCL0[ xSbIdx ][ ySbIdx ] and refMvCL1[ xSbIdx ][ ySbIdx ] with xSbIdx = 0..numSbX − 1, and ySbIdx = 0..numSbY − 1, the reference indices refIdxL0 and refIdxL1, the prediction list utilization flags predFlagL0[ xSbIdx ][ ySbIdx ] and predFlagL1[ xSbIdx ][ ySbIdx ], the half sample interpolation filter index hpelIfIdx, the bi-prediction weight index bcwIdx, the mimimum sum of absolute difference values in decoder-side motion vector refinement process dmvrSad[ xSbIdx ][ ySbIdx ], the decoder-side motion vector refinement flag dmvrFlag, the variable cIdx set equal to 1, the prediction refinement utility flags cbProfFlagL0 and cbProfFlagL1, and motion vector difference arrays diffMvL0[ xIdx ][ yIdx ] and diffMvL1[ xIdx ][ yIdx ] with xIdx = 0..cbWidth / numSbX − 1, and yIdx = 0..cbHeight / numSbY − 1 as inputs, and the inter prediction samples (predSamples) that are an (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamplesCb of prediction chroma samples for the chroma components Cb as outputs.
      * When sps\_chroma\_format\_idc is not equal to 0. the decoding process for inter blocks as specified in subclause 8.5.6.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight, the number of luma subblocks in horizontal direction numSbX and in vertical direction numSbY, the chroma motion vectors mvCL0[ xSbIdx ][ ySbIdx ] and mvCL1[ xSbIdx ][ ySbIdx ], and the refined chroma motion vectors refMvCL0[ xSbIdx ][ ySbIdx ] and refMvCL1[ xSbIdx ][ ySbIdx ] with xSbIdx = 0..numSbX − 1, and ySbIdx = 0..numSbY − 1, the reference indices refIdxL0 and refIdxL1, the prediction list utilization flags predFlagL0[ xSbIdx ][ ySbIdx ] and predFlagL1[ xSbIdx ][ ySbIdx ], the half sample interpolation filter index hpelIfIdx, the bi-prediction weight index bcwIdx, the mimimum sum of absolute difference values in decoder-side motion vector refinement process dmvrSad[ xSbIdx ][ ySbIdx ], the decoder-side motion vector refinement flag dmvrFlag, the variable cIdx set equal to 2, the prediction refinement utility flags cbProfFlagL0 and cbProfFlagL1, and motion vector difference arrays diffMvL0[ xIdx ][ yIdx ] and diffMvL1[ xIdx ][ yIdx ] with xIdx = 0..cbWidth / numSbX − 1, and yIdx = 0..cbHeight / numSbY − 1 as inputs, and the inter prediction samples (predSamples) that are an (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamplesCr of prediction chroma samples for the chroma components Cr as outputs.
* Otherwise (MergeGpmFlag[ xCb ][ yCb ] is equal to 1), the decoding process for geometric partitioning mode inter blocks as specified in subclause 8.5.7.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight, the luma motion vectors mvA and mvB, the chroma motion vectors mvCA and mvCB, the reference indices refIdxA and refIdxB, and the prediction list flags predListFlagA and predListFlagB as inputs, and the inter prediction samples (predSamples) that are an (cbWidth)x(cbHeight) array predSamplesL of prediction luma samples and, when sps\_chroma\_format\_idc is not equal to 0, two (cbWidth / SubWidthC)x(cbHeight / SubHeightC) arrays predSamplesCb and predSamplesCr of prediction chroma samples, one for each of the chroma components Cb and Cr, as outputs.

1. The variables NumSbX[ xCb ][ yCb ] and NumSbY[ xCb ][ yCb ] are set equal to numSbX and numSbY, respectively.
2. The residual samples of the current coding unit are derived as follows:

* The decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in subclause 8.5.8 is invoked with the location ( xTb0, yTb0 ) set equal to the luma location ( xCb, yCb ), the width nCbW set equal to the luma coding block width cbWidth, the height nCbH set equal to the luma coding block height cbHeight, the width nTbW set equal to the luma coding block width cbWidth, the height nTbH set equal to the luma coding block height cbHeight and the variable cIdx set equal to 0 as inputs, and the array resSamplesL as output.
* When sps\_chroma\_format\_idc is not equal to 0. the decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in subclause 8.5.8 is invoked with the location ( xTb0, yTb0 ) set equal to the chroma location ( xCb / SubWidthC, yCb / SubHeightC ), the width nCbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nCbH set equal to the chroma coding block height cbHeight / SubHeightC, the width nTbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nTbH set equal to the chroma coding block height cbHeight / SubHeightC and the variable cIdx set equal to 1 as inputs, and the array resSamplesCb as output.
* When sps\_chroma\_format\_idc is not equal to 0. the decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in subclause 8.5.8 is invoked with the location ( xTb0, yTb0 ) set equal to the chroma location ( xCb / SubWidthC, yCb / SubHeightC ), the width nCbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nCbH set equal to the chroma coding block height cbHeight / SubHeightC, the width nTbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nTbH set equal to the chroma coding block height cbHeight / SubHeightC and the variable cIdx set equal to 2 as inputs, and the array resSamplesCr as output.
* When cu\_act\_enabled\_flag[ xCb ][ yCb ] is equal to 1, the residual modification process for residual blocks using colour space conversion as specified in subclause 8.7.4.6 is invoked with the variable nTbW set equal to cbWidth, the variable nTbH set equal to cbHeight, the array rY set equal to resSamplesL, the array rCb set equal to resSamplesCb, and the array rCr set equal to resSamplesCr as inputs, and the output are modified versions of the arrays resSamplesL, resSamplesCb and resSamplesCr.

1. The reconstructed samples of the current coding unit are derived as follows:

* The picture reconstruction process for a colour component as specified in subclause 8.7.5.1 is invoked with the block location ( xCurr, yCurr ) set equal to ( xCb, yCb ), the block width nCurrSw set equal to cbWidth, the block height nCurrSh set equal to cbHeight, the variable cIdx set equal to 0, the (cbWidth)x(cbHeight) array predSamples set equal to predSamplesL and the (cbWidth)x(cbHeight) array resSamples set equal to resSamplesL as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* When sps\_chroma\_format\_idc is not equal to 0. the decoding process for the reconstructed signal of chroma coding blocks coded in inter prediction mode as specified in subclause 8.5.9 is invoked with the transform block location ( xTb0, yTb0 ) set equal to ( xCb / SubWidthC, yCb / SubHeightC ), the transform block width nTbW set equal to cbWidth / SubWidthC and the height nTbH set equal to cbHeight / SubHeightC, the variable cIdx set equal to 1, the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamples set equal to predSamplesCb and the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array resSamples set equal to resSamplesCb as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* When sps\_chroma\_format\_idc is not equal to 0. the decoding process for the reconstructed signal of chroma coding blocks coded in inter prediction mode as specified in subclause 8.5.9 is invoked with the transform block location ( xTb0, yTb0 ) set equal to ( xCb / SubWidthC, yCb / SubHeightC ), the transform block width nTbW set equal to cbWidth / SubWidthC and the height nTbH set equal to cbHeight / SubHeightC, the variable cIdx set equal to 2, the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamples set equal to predSamplesCr and the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array resSamples set equal to resSamplesCr as inputs, and the output is a modified reconstructed picture before in-loop filtering.

### Decoding process for the reconstructed signal of chroma coding blocks coded in inter prediction mode

Inputs to this process are:

* a sample location ( xTb0, yTb0 ) specifying the top-left sample of the current transform block relative to the top‑left sample of the current picture,
* a variable nTbW specifying the width of the current transform block,
* a variable nTbH specifying the height of the current transform block,
* a variable cIdx specifying the colour component of the current block,
* an (nTbW)x(nTbH) array predSamples specifying the prediction samples of the current block,
* an (nTbW)x(nTbH) array resSamples specifying the residual samples of the current block,

Output of this process is a modified reconstructed picture before in-loop filtering.

The maximum transform block width maxTbWidth and height maxTbHeight are derived as follows:

maxTbWidth = MaxTbSizeY / SubWidthC (1052)

maxTbHeight = MaxTbSizeY / SubHeightC (1053)

Depending on maxTbSize, the following applies:

* If nTbW is greater than maxTbWidth or nTbH is greater than maxTbHeight, the following ordered steps apply.

1. The variables verSplitFirst, newTbW and newTbH are derived as follows:

verSplitFirst = ( nTbW \* SubWidthC > nTbH \*SubHeightC ) && ( nTbW > maxTbWidth ) (1055)

newTbW = verSplitFirst ? ( nTbW / 2 ) : nTbW (1056)

newTbH = !verSplitFirst ? ( nTbH / 2 ) : nTbH (1057)

1. The decoding process for the reconstructed signal of chroma coding blocks coded in inter prediction mode as specified in this subclause is invoked with ( xTb0, yTb0 ), the transform block width nTbW set equal to newTbW and the height nTbH set equal to newTbH, cIdx, the (newTbW)x(newTbH) array predSamples set equal to predSamples[ x ][ y ], and the (newTbW)x(newTbH) array resSamples set equal to resSamples[ x ][ y ] with x = 0..newTbW − 1, y = 0..newTbH − 1 as inputs, and the output is a modified reconstructed picture before in-loop filtering.
2. The following applies:

* If verSplitFirst is equal to TRUE, the decoding process for the reconstructed signal of chroma coding blocks coded in inter prediction mode as specified in this subclause is invoked with the location ( xTb0, yTb0 ) set equal to ( xTb0 + newTbW, yTb0 ), the transform block width nTbW set equal to newTbW and the height nTbH set equal to newTbH, cIdx, the (newTbW)x(newTbH) array predSamples set equal to predSamples[ x ][ y ], and the (newTbW)x(newTbH) array resSamples set equal to resSamples[ x ][ y ] with x = newTbW..2 \* newTbW − 1, y = 0..newTbH − 1 as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* Otherwise (verSplitFirst is equal to FALSE), the process for the reconstructed signal of chroma coding blocks coded in inter prediction mode as specified in this subclause is invoked with the location ( xTb0, yTb0 ) set equal to ( xTb0, yTb0 + newTbH ), the transform block width nTbW set equal to newTbW and the height nTbH set equal to newTbH, cIdx, the (newTbW)x(newTbH) array predSamples set equal to predSamples[ x ][ y ], and the (newTbW)x(newTbH) array resSamples set equal to resSamples[ x ][ y ] with x = 0..newTbW − 1, y = newTbH..2 \* newTbH − 1 as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* Otherwise, if cu\_sbt\_flag is equal to 1, the following applies:
* The variables nTb0W, nTb0H, nTb1W, nTb1H, xTb1, and yTb1 are derived as follows:

nTb0W = !cu\_sbt\_horizontal\_flag ? ( nTbW \* SbtNumFourthsTb0 / 4 ) : nTbW (1059)

nTb0H = !cu\_sbt\_horizontal\_flag ? nTbH : ( nTbH \* SbtNumFourthsTb0 / 4 ) (1060)

nTb1W = nTbW − ( !cu\_sbt\_horizontal\_flag ? nTb0W : 0 ) (1059)

nTb1H = nTbH − ( !cu\_sbt\_horizontal\_flag ? 0 : nTb0H ) (1060)

xTb1 = xTb0 + ( !cu\_sbt\_horizontal\_flag ? nTb0W : 0 ) (1059)

yTb1 = yTb0 + ( !cu\_sbt\_horizontal\_flag ? 0 : nTb0H ) (1060)

* The picture reconstruction process for a colour component as specified in subclause 8.7.5.1 is invoked with the block location ( xCurr, yCurr ) set equal to ( xTb0, yTb0 ), the block width nCurrSw set equal to nTb0W, the block height nCurrSh set equal to nTb0H, cIdx, the (nTb0W)x(nTb0H) array predSamples set equal to predSamples[ x ][ y ], and the (nTb0W)x(nTb0H) array resSamples set equal to resSamples[ x ][ y ] with x = 0..nTb0W − 1, y = 0..nTb0H − 1 as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* The picture reconstruction process for a colour component as specified in subclause 8.7.5.1 is invoked with the block location ( xCurr, yCurr ) set equal to ( xTb1, yTb1 ), the block width nCurrSw set equal to nTb1W, the block height nCurrSh set equal to nTb1H, cIdx, the (nTb1W)x(nTb1H) array predSamples set equal to predSamples[ xTb1 − xTb0 + x ][ yTb1 − yTb0 + y ], and the (nTb1W)x(nTb1H) array resSamples set equal to resSamples[ xTb1 − xTb0 + x ][ yTb1 − yTb0 + y ] with x = 0..nTb1W − 1, y = 0..nTb1H − 1 as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* Otherwise, picture reconstruction process for a colour component as specified in subclause 8.7.5.1 is invoked with the block location ( xCurr, yCurr ) set equal to ( xTb0, yTb0 ), the block width nCurrSw set equal to nTbW, the block height nCurrSh set equal to nTbH, cIdx, the (nTbW)x(nTbH) array predSamples, and the (nTbW)x(nTbH) array resSamples as inputs, and the output is a modified reconstructed picture before in-loop filtering.

## Decoding process for coding units coded in IBC prediction mode

### General decoding process for coding units coded in IBC prediction mode

Inputs to this process are:

* a luma location ( xCb, yCb ) specifying the top-left sample of the current coding block relative to the top‑left luma sample of the current picture,
* a variable cbWidth specifying the width of the current coding block in luma samples,
* a variable cbHeight specifying the height of the current coding block in luma samples,
* a variable treeType specifying whether a single or a dual tree is used and if a dual tree is used, it specifies whether the current tree corresponds to the luma or chroma components.

Output of this process is a modified reconstructed picture before in-loop filtering.

The derivation process for quantization parameters as specified in subclause 8.7.1 is invoked with the luma location ( xCb, yCb ), the width of the current coding block in luma samples cbWidth and the height of the current coding block in luma samples cbHeight, and the variable treeType as inputs.

The variable IsGt4by4 is derived as follows:

IsGt4by4 = ( cbWidth \* cbHeight ) > 16 (1070)

The decoding process for coding units coded in IBC prediction mode consists of the following ordered steps:

1. The block vector components of the current coding unit are derived as follows:

* The derivation process for block vector components as specified in subclause 8.6.2.1 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight as inputs, and the luma block vector bvL as output.
* When treeType is equal to SINGLE\_TREE, the derivation process for chroma block vectors in subclause 8.6.2.5 is invoked with luma block vector bvL as input, and chroma block vector bvC as output.

1. The prediction samples of the current coding unit are derived as follows:

* The decoding process for IBC blocks as specified in subclause 8.6.3 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight, the luma block vector bvL, the variable cIdx set equal to 0 as inputs, and the IBC prediction samples (predSamples) that are an (cbWidth)x(cbHeight) array predSamplesL of prediction luma samples as outputs.
* When treeType is equal to SINGLE\_TREE, the prediction samples of the current coding unit are derived as follows:
  + - * The decoding process for IBC blocks as specified in subclause 8.6.3 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight, the chroma block vector bvC and the variable cIdx set equal to 1 as inputs, and the IBC prediction samples (predSamples) that are an (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamplesCb of prediction chroma samples for the chroma components Cb as outputs.
      * The decoding process for IBC blocks as specified in subclause 8.6.3 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block width cbWidth and the luma coding block height cbHeight, the chroma block vector bvC and the variable cIdx set equal to 2 as inputs, and the IBC prediction samples (predSamples) that are an (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamplesCr of prediction chroma samples for the chroma components Cr as outputs.

1. The residual samples of the current coding unit are derived as follows:

* The decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in subclause 8.5.8 is invoked with the location ( xTb0, yTb0 ) set equal to the luma location ( xCb, yCb ), the width nCbW set equal to the luma coding block width cbWidth, the height nCbH set equal to the luma coding block height cbHeight, the width nTbW set equal to the luma coding block width cbWidth, the height nTbH set equal to the luma coding block height cbHeight and the variable cIdx set equal to 0 as inputs, and the array resSamplesL as output.
* When treeType is equal to SINGLE\_TREE, the decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in subclause 8.5.8 is invoked with the location ( xTb0, yTb0 ) set equal to the chroma location ( xCb / SubWidthC, yCb / SubHeightC ), the width nCbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nCbH set equal to the chroma coding block height cbHeight / SubHeightC, the width nTbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nTbH set equal to the chroma coding block height cbHeight / SubHeightC and the variable cIdx set equal to 1 as inputs, and the array resSamplesCb as output.
* When treeType is equal to SINGLE\_TREE, the decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in subclause 8.5.8 is invoked with the location ( xTb0, yTb0 ) set equal to the chroma location ( xCb / SubWidthC, yCb / SubHeightC ), the width nCbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nCbH set equal to the chroma coding block height cbHeight / SubHeightC, the width nTbW set equal to the chroma coding block width cbWidth / SubWidthC, the height nTbH set equal to the chroma coding block height cbHeight / SubHeightC and the variable cIdxset equal to 2 as inputs, and the array resSamplesCr as output.
* When cu\_act\_enabled\_flag[ xCb ][ yCb ] is equal to 1, the residual modification process for residual blocks using colour space conversion as specified in subclause 8.7.4.6 is invoked with the variable nTbW set equal to cbWidth, the variable nTbH set equal to cbHeight, the array rY set equal to resSamplesL, the array rCb set equal to resSamplesCb, and the array rCr set equal to resSamplesCr as inputs, and the output are modified versions of the arrays resSamplesL, resSamplesCb and resSamplesCr.

1. The reconstructed samples of the current coding unit are derived as follows:

* The picture reconstruction process for a colour component as specified in subclause 8.7.5.1 is invoked with the block location ( xCurr, yCurr ) set equal to ( xCb, yCb ), the block width nCurrSw set equal to cbWidth, the block height nCurrSh set equal to cbHeight, the variable cIdx set equal to 0, the (cbWidth)x(cbHeight) array predSamples set equal to predSamplesL and the (cbWidth)x(cbHeight) array resSamples set equal to resSamplesL as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* When treeType is equal to SINGLE\_TREE, the decoding process for the reconstructed signal of chroma coding blocks coded in inter prediction mode as specified in subclause 8.5.9 is invoked with the transform block location ( xTb0, yTb0 ) set equal to ( xCb / SubWidthC, yCb / SubHeightC ), the transform block width nTbW set equal to cbWidth / SubWidthC and the height nTbH set equal to cbHeight / SubHeightC, the variable cIdx set equal to 1, the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamples set equal to predSamplesCb and the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array resSamples set equal to resSamplesCb as inputs, and the output is a modified reconstructed picture before in-loop filtering.
* When treeType is equal to SINGLE\_TREE, the decoding process for the reconstructed signal of chroma coding blocks coded in inter prediction mode as specified in subclause 8.5.9 is invoked with the transform block location ( xTb0, yTb0 ) set equal to ( xCb / SubWidthC, yCb / SubHeightC ), the transform block width nTbW set equal to cbWidth / SubWidthC and the height nTbH set equal to cbHeight / SubHeightC, the variable cIdx set equal to 2, the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array predSamples set equal to predSamplesCr and the (cbWidth / SubWidthC)x(cbHeight / SubHeightC) array resSamples set equal to resSamplesCr as inputs, and the output is a modified reconstructed picture before in-loop filtering.