Revision r1

WD changes for JCTVC-S0177 based on JCTVC-R1005-v3:

### Decoding process for intra blocks

#### General decoding process for intra blocks

…

– Otherwise (splitFlag is equal to 0), for the variable blkIdx proceeding over the values 0..( cIdx > 0  &&  ChromaArrayType  = =  2 ? 1 : 0 ), the following ordered steps apply:

1. The variable nTbS is set equal to 1  <<  log2TrafoSize.
2. The variable yTbOffset is set equal to blkIdx \* nTbS.
3. The variable yTbOffsetY is set equal to yTbOffset \* SubHeightC.
4. When controlParaACT is not equal to 2, the variable residualDpcm is derived as follows:
   * If all of the following conditions are true, residualDpcm is set equal to 1.
   * implicit\_rdpcm\_enabled\_flag is equal to 1.
   * either transform\_skip\_flag[ xTbY ][ yTbY + yTbOffsetY ][ cIdx ] is equal to 1, or cu\_transquant\_bypass\_flag is equal to 1.
   * either predModeIntra is equal to 10, or predModeIntra is equal to 26.
   * Otherwise, residualDpcm is set equal to explicit\_rdpcm\_flag[ xTbY ][ yTbY + yTbOffsetY ][ cIdx ].
5. When controlParaACT is not equal to 1, depending upon the value of predModeIntraBc, the following applies:

– When predModeIntraBc is equal to 0, the general intra sample prediction process as specified in subclause 8.4.4.2.1 is invoked with the transform block location ( xTb0, yTb0 + yTbOffset ), the intra prediction mode predModeIntra, the transform block size nTbS, and the variable cIdx as inputs, and the output is an (nTbS)x(nTbS) array predSamples and a 1x(2\*nTbs-1) array predSampleDiff.

– Otherwise (predModeIntraBc is equal to 1), the intra block copying process as specified in subclause 8.4.4.2.7 is invoked with the transform block location ( xTb0, yTb0 + yTbOffset ), the transform block size nTbS, the variable trafoDepth, the variable bvIntra, and the variable cIdx as inputs, and the output is an (nTbS)x(nTbS) array predSamples.

1. When controlParaACT is not equal to 2, the scaling and transformation process as specified in subclause 8.6.2 is invoked with the luma location ( xTbY, yTbY + yTbOffsetY ), the variable trafoDepth, the variable cIdx, and the transform size trafoSize set equal to nTbS as inputs, and the output is an (nTbS)x(nTbS) array resSamples.
2. When controlParaACT is not equal to 2 and residualDpcm is equal to 1, the directional residual modification process for blocks using a transform bypass as specified in subclause 8.6.5 is invoked with the variable mDir set equal to predModeIntra / 26, the variable nTbS, and the (nTbS)x(nTbS) array r set equal to the array resSamples as inputs, and the output is a modified (nTbS)x(nTbS) array resSamples.
3. When controlParaACT is not equal to 2 and cross\_component\_prediction\_enabled\_flag is equal to 1, ChromaArrayType is equal to 3, and cIdx is not equal to 0, the residual modification process for transform blocks using cross-component prediction as specified in subclause 8.6.6 is invoked with the current luma transform block location ( xTbY, yTbY ), the variable nTbS, the variable cIdx, the (nTbS)x(nTbS) array rY set equal to the corresponding luma residual sample array resSamples of the current transform block, the 1x(2\*nTbs-1) array b set equal to array predSampleDiff, and the (nTbS)x(nTbS) array r set equal to the array resSamples as inputs, and the output is a modified (nTbS)x(nTbS) array resSamples.
4. When controlParaACT is not equal to 1, the picture construction process prior to in-loop filtering for a colour component as specified in subclause 8.6.7 is invoked with the transform block location ( xTb0, yTb0 + yTbOffset ), the variables nCurrSw and nCurrSh both set equal to nTbS, the variable cIdx, the (nTbS)x(nTbS) array predSamples, and the (nTbS)x(nTbS) array resSamples as inputs.

…

**8.4.5.2.5 Specification of intra prediction mode INTRA\_DCInputs to this process are:**

– the neighbouring samples p[ x ][ y ], with x = −1, y = −1..nTbS \* 2 − 1 and x = 0..nTbS \* 2 − 1, y = −1,

– a variable nTbS specifying the transform block size,

– a variable cIdx specifying the colour component of the current block.

Outputs of this process are

– the predicted samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1,

– the offset values predSampleDiff[m], with m = 0..2\*nTbS – 2.

The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1, are derived by the following ordered steps:

1. A variable dcVal is derived as follows:

dcVal =  (8‑41)

where k = Log2( nTbS ).

1. Depending on the value of the colour component index cIdx, the following applies:

* If cIdx is equal to 0 and nTbS is less than 32, the following applies:

predSamples[ 0 ][ 0 ] = ( p[ −1 ][ 0 ] + 2 \* dcVal + p[ 0 ][ −1 ] + 2 )  >>  2 (8‑42)

predSamples[ x ][ 0 ] = ( p[ x ][ −1 ] + 3 \* dcVal + 2 )  >>  2, with x = 1..nTbS − 1 (8‑43)

predSamples[ 0 ][ y ] = ( p[ −1 ][ y ] + 3 \* dcVal + 2 )  >>  2, with y = 1..nTbS − 1 (8‑44)

predSamples[ x ][ y ] = dcVal, with x, y = 1..nTbS − 1 (8‑45)

* Otherwise, the prediction samples predSamples[ x ][ y ] are derived as follows:

predSamples[ x ][ y ] = dcVal, with x, y = 0..nTbS − 1 (8‑46)

1. The offset values predSampleDiff[ m ] are derived as follows:

predSampleDiff[ m ] = predSamples[ x ][ y ] − dcVal, with x=0, y= 0..nTbS – 1 and x= 1..nTbS – 1, y=0 for m = nTbS – 1 – y + x.

* + - * 1. **Specification of intra prediction mode in the range of INTRA\_ANGULAR2.. INTRA\_ANGULAR34**

Inputs to this process are:

– the intra prediction mode predModeIntra,

– the neighbouring samples p[ x ][ y ], with x = −1, y = −1..nTbS \* 2 − 1 and x = 0..nTbS \* 2 − 1, y = −1,

– a variable nTbS specifying the transform block size,

– a variable cIdx specifying the colour component of the current block.

Outputs of this process are

– the predicted samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1,

– the offset values predSampleDiff[m], with m = 0..2\*nTbS – 2.

…

The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:

– If predModeIntra is greater than or equal to 18, the following ordered steps apply:

1. The reference sample array ref[ x ] is specified as follows:

* The following applies:

ref[ x ] = p[ −1 + x ][ −1 ], with x = 0..nTbS (8‑47)

* If intraPredAngle is less than 0, the main reference sample array is extended as follows:
* When ( nTbS \* intraPredAngle )  >>  5 is less than −1,

ref[ x ] = p[ −1 ][ −1 + ( ( x \* invAngle + 128 )  >>  8 ) ],  
 with x = −1..( nTbS \* intraPredAngle )  >>  5 (8‑48)

* Otherwise,

ref[ x ] = p[ −1 + x ][ −1 ], with x = nTbS + 1..2 \* nTbS (8‑49)

1. The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:
   1. The index variable iIdx and the multiplication factor iFact are derived as follows:

iIdx = ( ( y + 1 ) \* intraPredAngle )  >>  5 (8‑50)

iFact = ( ( y + 1 ) \* intraPredAngle ) & 31 (8‑51)

* 1. Depending on the value of iFact, the following applies:
* If iFact is not equal to 0, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] =   
 ( ( 32 − iFact ) \* ref[ x + iIdx + 1 ] + iFact \* ref[ x + iIdx + 2 ] + 16 )  >>  5 (8‑52)

* Otherwise, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] = ref[ x + iIdx + 1 ] (8‑53)

* 1. ~~When~~ If predModeIntra is equal to 26 (vertical), cIdx is equal to 0, nTbS is less than 32, and disableIntraBoundaryFilter is equal to 0, then the following filtering applies with x = 0, y = 0..nTbS − 1:

predSamples[ x ][ y ] = Clip1Y( p[ x ][ −1 ] + ( ( p[ −1 ][ y ] − p[ −1 ][ −1 ] )  >>  1 ) ) (8‑54)

and the offset values predSampleDiff[ m ] are derived as follows:

predSampleDiff[  m ] = predSamples[ 0 ][ y ] − p[ x ][ −1 ] , with x=0, y= 0..nTbS – 1 for m = nTbS – 1 – y + x.

predSampleDiff[ m ] = 0 , with m = nTbS..2\*nTbS – 2.

– Otherwise the offset values predSampleDiff[ m ] are derived as follows:

predSampleDiff[ m ] = 0 , with m = 0..2\*nTbS – 2.

– Otherwise (predModeIntra is less than 18), the following ordered steps apply:

1. The reference sample array ref[ x ] is specified as follows:

* The following applies:

ref[ x ] = p[ −1 ][ −1 + x ], with x = 0..nTbS (8‑55)

* If intraPredAngle is less than 0, the main reference sample array is extended as follows:
* When ( nTbS \* intraPredAngle )  >>  5 is less than −1,

ref[ x ] = p[ −1 + ( ( x \* invAngle + 128 )  >>  8 ) ][ −1 ],  
 with x = −1..( nTbS \* intraPredAngle )  >>  5 (8‑56)

* Otherwise,

ref[ x ] = p[ −1 ][ −1 + x ], with x = nTbS + 1..2 \* nTbS (8‑57)

1. The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:
2. The index variable iIdx and the multiplication factor iFact are derived as follows:

iIdx = ( ( x + 1 ) \* intraPredAngle )  >>  5 (8‑58)

iFact = ( ( x + 1 ) \* intraPredAngle ) & 31 (8‑59)

1. Depending on the value of iFact, the following applies:

* If iFact is not equal to 0, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] =   
 ( ( 32 − iFact ) \* ref[ y + iIdx + 1 ] + iFact \* ref[ y + iIdx + 2 ] + 16 )  >>  5 (8‑60)

* Otherwise, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] = ref[ y + iIdx + 1 ] (8‑61)

1. ~~When~~ If predModeIntra is equal to 10 (horizontal), cIdx is equal to 0, nTbS is less than 32, and disableIntraBoundaryFilter is equal to 0, then the following filtering applies with x = 0..nTbS − 1, y = 0:

predSamples[ x ][ y ] = Clip1Y( p[ −1 ][ y ] + ( ( p[ x ][ −1 ] − p[ −1 ][ −1 ] )  >>  1 ) ) (8‑62)

and the offset values predSampleDiff[ m ] are derived as follows:

predSampleDiff[  m ] = predSamples[ x ][ 0 ] − p[ −1 ][ y ] , with x=0..nTbS – 1, y= 0 for m = nTbS – 1 – y + x.

predSampleDiff[ m ] = 0 , with m = 0..nTbS – 2.

– Otherwise the offset values predSampleDiff[ m ] are derived as follows:

predSampleDiff[ m ] = 0 , with m = 0..2\*nTbS – 2.

**8.6.6 Residual modification process for transform blocks using cross-component prediction**

This process is only invoked when ChromaArrayType is equal to 3.

Inputs to this process are:

– a luma location ( xTbY, yTbY ) specifying the top-left sample of the current luma transform block relative to the top‑left luma sample of the current picture,

– a variable nTbS specifying the transform block size,

– a variable cIdx specifying the colour component of the current block,

– an (nTbS)x(nTbS) array of luma residual samples rY with elements rY[ x ][ y ],

– a 1x(2\*nTbS-1) array of prediction offset samples b with elements b[m].

– a cu\_residual\_act\_flag[ xTbY ][ yTbY ]

– an (nTbS)x(nTbS) array of residual samples r with elements r[ x ][ y ].

Output of this process is the modified (nTbS)x(nTbS) array r of residual samples.

The (nTbS)x(nTbS) array of residual samples r with  x = 1~~0~~..nTbS − 1, y = 1~~0~~..nTbS − 1  is modified as follows:

r[ x ][ y ] += ( ResScaleVal[ cIdx ][ xTbY ][ yTbY ] \* (8‑294)  
 ( ( rY[ x ][ y ]  <<  BitDepthC )  >>  BitDepthY ) ) >> 3

r with x = 0, y = 0..nTbS − 1 and  r with x = 1..nTbS − 1 , y = 0 is modified as follows:

if cu\_residual\_act\_flag[ xTbY ][ yTbY ] is equal to 0, then

r[ x ][ y ] += ( ResScaleVal[ cIdx ][ xTbY ][ yTbY ] \* (8‑299)  
 (  ( ( rY[ x ][ y ] + b[ nTbS – 1 − y + x ])  <<  BitDepthC )  >>  BitDepthY ) ) >> 3

otherwise,

if cIdx is equal to 1, then

r[ x ][ y ] += ( ResScaleVal[ cIdx ][ xTbY ][ yTbY ] \* (8‑300)  
 (  ( ( rY[ x ][ y ] + b[ nTbS – 1 − y + x ] >> 2)  <<  BitDepthC )  >>  BitDepthY ) ) >> 3

otherwise, if cIdx is equal to 2, then

r[ x ][ y ] += ( ResScaleVal[ cIdx ][ xTbY ][ yTbY ] \* (8‑301)  
 (  ( ( rY[ x ][ y ] + b[ nTbS – 1 − y + x ] >> 2)  <<  BitDepthC )  >>  BitDepthY ) ) >> 3 - b[ nTbS – 1 − y + x ] >> 2