# Draft Text Specification

The proposed text changes are based on the document JCTVC-M1005-v2.doc. The changes are marked in yellow.

**1 Changes to residual coding syntax and semantics**

**7.3.8.11 Residual coding syntax**

|  |  |
| --- | --- |
| residual\_coding( x0, y0, log2TrafoSize, cIdx ) { | **Descriptor** |
| if( transform\_skip\_enabled\_flag && !cu\_transquant\_bypass\_flag &&   ( log2TrafoSize = = 2 ) ) |  |
| **transform\_skip\_flag**[ x0 ][ y0 ][ cIdx ] | ae(v) |
| if( transform\_skip\_flag[ x0 ][ y0 ][ cIdx ] &&   CuPredMode[ x0 ][ y0 ] == MODE\_INTER ) { |  |
| **inter\_rdpcm\_flag**[ x0 ][ y0 ][ cIdx ] | ae(v) |
| if( inter\_rdpcm\_flag[ x0 ][ y0 ][ cIdx ] ) |  |
| **inter\_rdpcm\_dir\_flag**[ x0 ][ y0 ][ cIdx ] | ae(v) |
| } |  |
| **coeff\_abs\_level\_greater1\_flag**[ n ] | ae(v) |
| numGreater1Flag++ |  |
| if( coeff\_abs\_level\_greater1\_flag[ n ] && lastGreater1ScanPos = = −1 ) |  |
| lastGreater1ScanPos = n |  |
| } |  |
| if( lastSigScanPos = = −1 ) |  |
| lastSigScanPos = n |  |
| firstSigScanPos = n |  |
| } |  |
| } |  |
| signHidden = ( lastSigScanPos − firstSigScanPos > 3 && !cu\_transquant\_bypass\_flag ) |  |
| if ( transform\_skip\_flag[ x0 ] [ y0 ] [ cIdx ] && ( ( CuPredMode[ x0 ][ y0 ] == MODE\_INTRA ) && ( ( predModeIntra == 26 ) || ( predModeIntra == 10 ) ) ) || ( CuPredMode[ x0 ][ y0 ] == MODE\_INTER && inter\_rdpcm\_flag[ x0 ][ y0 ][ cIdx ] ) ) |  |
| signHidden = 0 |  |
| if( lastGreater1ScanPos != −1 ) |  |
| **coeff\_abs\_level\_greater2\_flag**[ lastGreater1ScanPos ] | ae(v) |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ( xS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 0 ] |  |
| yC = ( yS << 2 ) + ScanOrder[ 2 ][ scanIdx ][ n ][ 1 ] |  |
| if( sig\_coeff\_flag[ xC ][ yC ] &&   ( !sign\_data\_hiding\_enabled\_flag | | !signHidden | | ( n != firstSigScanPos ) ) ) |  |
| **coeff\_sign\_flag**[ n ] | ae(v) |
| } |  |
| … |  |

**7.4.9.10 Residual coding semantics**

...

**inter\_rdpcm\_flag**[ x0 ][ y0 ][ cIdx ] indicates whether the inter residuals prediction process specified in clause 8.6.6 is applied to the associated transform block or not. The array indices x0, y0 specify the location ( x0, y0 ) of the top-left luma sample of the considered transform block relative to the top-left luma sample of the picture. The array index cIdx specifies an indicator for the colour component; it is equal to 0 for luma, equal to 1 for Cb, and equal to 2 for Cr. inter\_rdpcm\_flag[ x0 ][ y0 ][ cIdx ] equal to 1 specifies that the process indicated in clause 8.6.6 is applied to the current transform block. inter\_rdpcm\_flag [ x0 ][ y0 ][ cIdx ] equal to 0 specifies that the process indicated in clause 8.6.6 is not applied to the current transform block. When inter\_rdpcm\_flag[ x0 ][ y0 ][ cIdx ] is not present, it is inferred to be equal to 0.

**inter\_rdpcm\_dir\_flag**[ x0 ][ y0 ][ cIdx ] indicates the direction where the inter residuals prediction process specified in clause 8.6.6 is applied. The array indices x0, y0 specify the location ( x0, y0 ) of the top-left luma sample of the considered transform block relative to the top-left luma sample of the picture. The array index cIdx specifies an indicator for the colour component; it is equal to 0 for luma, equal to 1 for Cb, and equal to 2 for Cr. inter\_rdpcm\_dir\_flag[ x0 ][ y0 ][ cIdx ] equal to 1 specifies that the direction where the process specified in clause 8.6.6 is vertical. inter\_rdpcm\_dir\_flag[ x0 ][ y0 ][ cIdx ] equal to 1 specifies that the direction where the process specified in clause 8.6.6 is horizontal.

...

**3 Changes to decoding process for intra blocks**

**8.4.4** **Decoding process for intra blocks**

**8.4.4.1** **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 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.
4. The scaling and transformation process as specified in subclause 8.6.2 is invoked with the luma location ( xTbY, yTbY + yTbOffset \* SubHeightC ), 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.
5. When either cu\_transquant\_bypass\_flag is equal to 1 or transform\_skip\_flag[ xTb0 ] [ yTb0 + yTbOffset ] [ cIdx ] is equal to 1 and either predModeIntra is equal to 10, or predModeIntra is equal to 26, the directional residual modification process for intra blocks as specified in subclause 8.4.4.3 is invoked with the intra prediction mode predModeIntra, 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.

The picture reconstruction process prior to in-loop filtering for a colour component as specified in subclause 8.6.5 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.

…

**4 Changes to scaling and transform process at transform unit level**

**8.6.2 Scaling and transformation process**

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 trafoDepth specifying the hierarchy depth of the current block relative to the coding block,

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

– a variable nTbS specifying the size of the current transform block.

Output of this process is the (nTbS)x(nTbS) array of residual samples r with elements r[ x ][ y ].

The quantization parameter qP is derived as follows:

– If cIdx is equal to 0,

qP = Qp′Y (8‑264)

– Otherwise, if cIdx is equal to 1,

qP = Qp′Cb (8‑265)

– Otherwise (cIdx is equal to 2),

qP = Qp′Cr (8‑266)

The (nTbS)x(nTbS) array of residual samples r is derived as follows:

* If cu\_transquant\_bypass\_flag is equal to 1, the (nTbS)x(nTbS) array r is set equal to the (nTbS)x(nTbS) array of transform coefficients TransCoeffLevel[ xTbY ][ yTbY ][ cIdx ].
* If transform\_skip\_flag[ xTbY ][ yTbY ][ cIdx ] is equal to 1 and CuPredMode[ xTbY ][ yTbY ] is equal to MODE\_INTER, the following ordered steps apply:
* The scaling process for transform coefficients as specified in subclause 8.6.3 is invoked with the transform block location ( xTbY, yTbY ), the size of the transform block nTbS, the colour component variable cIdx, and the quantization parameter qP as inputs, and the output is an (nTbS)x(nTbS) array of scaled transform coefficients d.
* If inter\_rdpcm\_flag[ xTbY ][ yTbY ][ cIdx ] is equal to 1, the inter residual transform-bypass decoding process as specified in subclause 8.6.6 is invoked with the size of the transform block nTbS, the inter\_rdpcm\_dir\_flag[ xTbY ][ yTbY ][ cIdx ], and the (nTbS)x(nTbS) array d as the inputs, and the output is the (nTbS)x(nTbS) array of residual samples r.
* Otherwise, the following ordered steps apply:

1. The scaling process for transform coefficients as specified in subclause 8.6.3 is invoked with the transform block location ( xTbY, yTbY ), the size of the transform block nTbS, the colour component variable cIdx, and the quantization parameter qP as inputs, and the output is an (nTbS)x(nTbS) array of scaled transform coefficients d.
2. The (nTbS)x(nTbS) array of residual samples r is derived as follows:

* If transform\_skip\_flag[ xTbY ][ yTbY ][ cIdx ] is equal to 1, the residual sample array values r[ x ][ y ] with x = 0..nTbS − 1, y = 0..nTbS − 1 are derived as follows:

r[ x ][ y ] = ( d[ x ][ y ]  <<  7 ) (8‑267)

* Otherwise (transform\_skip\_flag[ xTbY ][ yTbY ][ cIdx ] is equal to 0), the transformation process for scaled transform coefficients as specified in subclause 8.6.4 is invoked with the transform block location ( xTbY, yTbY ), the size of the transform block nTbS, the colour component variable cIdx, and the (nTbS)x(nTbS) array of scaled transform coefficients d as inputs, and the output is an (nTbS)x(nTbS) array of residual samples r.

1. The variable bdShift is derived as follows:

bdShift = ( cIdx = = 0 ) ? 20 − BitDepthY : 20 − BitDepthC (8‑268)

1. The residual sample values r[ x ][ y ] with x = 0..nTbS − 1, y = 0..nTbS − 1 are modified as follows:

r[ x ][ y ] = ( r[ x ][ y ] + ( 1  <<  ( bdShift − 1 ) ) ) >> bdShift (8‑269)

**5 Changes required by the inter residual DPCM processing**

**8.6.6 Inter residual transform-skip decoding process**

This process is invoked when transform\_skip\_flag[ xTbY ][ yTbY ][ cIdx ] is equal to 1, CuPredMode[ xCb ][ yCb ] is not equal to MODE\_INTRA, and the flag inter\_rdpcm\_flag[ xCb ][ yCb ][ cIdx ] is equal to 1 for the considered value of the cIdx array index. The process for the Cb and Cr components is applied in the same way as for the luma component.

Inputs to this process are:

– a variable nTbS specifying the transform block size,

− a variable inter\_rdpcm\_dir\_flag specifying the direction where the inter residuals prediction process should is applied,

− an (nTbS)x(nTbS) array d with elements d[ x ][ y ] which is either an array relating to a residual transform-bypass block of the luma component or an array relating to a residual transform-bypass block of the Cb and Cr component.

Output of this process is the (nTbS)x(nTbS) array of residual samples r with elements r[ x ][ y ] containing the results of the inter residual transform-bypass decoding process.

Let f be a temporary (nTbS)x(nTbS) array with elements f[ x ][ y ], which are derived by:

f[ x ][ y ] = d[ x ][ y ] with x=0.. nTbS -1 and y=0.. nTbS -1

Depending on the inter\_rdpcm\_dir\_flag, the following apply:

If inter\_rdpcm\_dir\_flag is equal to 0 (horizontal direction) the array d is modified as follows:

d[ x ][ y ] =  with x=0..nCurrSw -1 and y=0..nCurrSh -1

Otherwise (inter\_rdpcm\_dir\_flag is equal to 1, vertical direction) the array d is modified as follows:

d[ x ][ y ] =  with x=0..nCurrSw -1 and y=0..nCurrSh -1

Finally the elements r[ x ][ y ] are derived as follows:

r[ x ][ y ] = ( d[ x ][ y ]  <<  7 ) with x=0..nCurrSw -1 and y=0..nCurrSh -1

**6 Changes required to the initialization process for context variables**

The following changes to Table 9‑4 are required for the introduction of the inter\_rdpcm\_flag and inter\_rdpcm\_dir\_flag binary flags. Moreover, one initialization table (Table 9−X) is added.

**Table 9‑4 – Association of ctxIdx and syntax elements for each initializationType in the initialization process**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Syntax structure** | **Syntax element** | **ctxTable** | **initType** | | |
| **0** | **1** | **2** |
| sao( ) | sao\_merge\_left\_flag sao\_merge\_up\_flag | Table 9‑5 | 0 | 1 | 2 |
| sao\_type\_idx\_luma sao\_type\_idx\_chroma | Table 9‑6 | 0 | 1 | 2 |
| coding\_quadtree( ) | split\_cu\_flag[ ][ ] | Table 9‑7 | 0..2 | 3..5 | 6..8 |
| coding\_unit( ) | cu\_transquant\_bypass\_flag | Table 9‑8 | 0 | 1 | 2 |
| cu\_skip\_flag | Table 9‑9 |  | 0..2 | 3..5 |
| pred\_mode\_flag | Table 9‑10 |  | 0 | 1 |
| part\_mode | Table 9‑11 | 0 | 1..4 | 5..8 |
| prev\_intra\_luma\_pred\_flag[ ][ ] | Table 9‑12 | 0 | 1 | 2 |
| intra\_chroma\_pred\_mode[ ][ ] | Table 9‑13 | 0 | 1 | 2 |
| rqt\_root\_cbf | Table 9‑14 |  | 0 | 1 |
| prediction\_unit( ) | merge\_flag[ ][ ] | Table 9‑15 |  | 0 | 1 |
| merge\_idx[ ][ ] | Table 9‑16 |  | 0 | 1 |
| inter\_pred\_idc[ ][ ] | Table 9‑17 |  | 0..4 | 5..9 |
| ref\_idx\_l0[ ][ ], ref\_idx\_l1[ ][ ] | Table 9‑18 |  | 0..1 | 2..3 |
| mvp\_l0\_flag[ ][ ], mvp\_l1\_flag[ ][ ] | Table 9‑19 |  | 0 | 1 |
| transform\_tree( ) | split\_transform\_flag[ ][ ][ ] | Table 9‑20 | 0..2 | 3..5 | 6..8 |
| cbf\_luma[ ][ ][ ] | Table 9‑21 | 0..1 | 2..3 | 4..5 |
| cbf\_cb[ ][ ][ ], cbf\_cr[ ][ ][ ] | Table 9‑22 | 0..3 | 4..7 | 8..11 |
| mvd\_coding( ) | abs\_mvd\_greater0\_flag[ ] | Table 9‑23 |  | 0 | 2 |
| abs\_mvd\_greater1\_flag[ ] | Table 9‑23 |  | 1 | 3 |
| transform\_unit( ) | cu\_qp\_delta\_abs | Table 9‑24 | 0..1 | 2..3 | 4..5 |
| inter\_rdpcm\_flag[ ][ ][ 0 ] | Table 9−X |  | 0 | 1 |
| inter\_rdpcm\_flag[ ][ ][ 1 ]  inter\_rdpcm\_flag[ ][ ][ 2 ] | Table 9−X |  | 2 | 3 |
| inter\_rdpcm\_dir\_flag[ ][ ][ 0 ] | Table 9−X |  | 0 | 1 |
| inter\_rdpcm\_dir\_flag[ ][ ][ 1 ]  inter\_rdpcm\_dir\_flag[ ][ ][ 2 ] | Table 9−X |  | 2 | 3 |
| residual\_coding( ) | transform\_skip\_flag[ ][ ][ 0 ] | Table 9‑25 | 0 | 1 | 2 |
| transform\_skip\_flag[ ][ ][ 1 ] transform\_skip\_flag[ ][ ][ 2 ] | Table 9‑25 | 3 | 4 | 5 |
| last\_sig\_coeff\_x\_prefix | Table 9‑26 | 0..17 | 18..35 | 36..53 |
| last\_sig\_coeff\_y\_prefix | Table 9‑27 | 0..17 | 18..35 | 36..53 |
| coded\_sub\_block\_flag[ ][ ] | Table 9‑28 | 0..3 | 4..7 | 8..11 |
| sig\_coeff\_flag[ ][ ] | Table 9‑29 | 0..41 | 42..83 | 84..125 |
| coeff\_abs\_level\_greater1\_flag[ ] | Table 9‑30 | 0..23 | 24..47 | 48..71 |
| coeff\_abs\_level\_greater2\_flag[ ] | Table 9‑31 | 0..5 | 6..11 | 12..17 |

**Table 9‑X – Values of initValue for ctxIdx of inter\_rdpcm\_flag and inter\_rdpcm\_dir\_flag**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Initialization variable** | **ctxIdx of inter\_rdpcm\_flag and inter\_rdpcm\_dir\_flag** | | | |
| **0** | **1** | **2** | **3** |
| **initValue** | 139 | 139 | 139 | 139 |

**7 Changes required to the binarization process**

Table 9‑32 is changed to define the binarization process for the inter\_rdpcm\_flag and inter\_rdpcm\_dir\_flag flags.

| **Table 9‑32 – Syntax elements and associated binarizations** | | | |
| --- | --- | --- | --- |
| **Syntax structure** | **Syntax element** | **Binarization** | |
| **Process** | **Input parameters** |
| slice\_segment\_data( ) | end\_of\_slice\_segment\_flag | FL | cMax = 1 |
| end\_of\_sub\_stream\_one\_bit | FL | cMax = 1 |
| sao( ) | sao\_merge\_left\_flag | FL | cMax = 1 |
| sao\_merge\_up\_flag | FL | cMax = 1 |
| sao\_type\_idx\_luma | TR | cMax = 2, cRiceParam = 0 |
| sao\_type\_idx\_chroma | TR | cMax = 2, cRiceParam = 0 |
| sao\_offset\_abs[ ][ ][ ][ ] | TR | cMax = ( 1  <<  ( Min( bitDepth, 10 ) − 5 ) ) − 1, cRiceParam = 0 |
| sao\_offset\_sign[ ][ ][ ][ ] | FL | cMax = 1 |
| sao\_band\_position[ ][ ][ ] | FL | cMax = 31 |
| sao\_eo\_class\_luma | FL | cMax = 3 |
| sao\_eo\_class\_chroma | FL | cMax = 3 |
| coding\_quadtree( ) | split\_cu\_flag[ ][ ] | FL | cMax = 1 |
| coding\_unit( ) | cu\_transquant\_bypass\_flag | FL | cMax = 1 |
| cu\_skip\_flag | FL | cMax = 1 |
| pred\_mode\_flag | FL | cMax = 1 |
| part\_mode | 9.3.3.5 | ( xCb, yCb ) = ( x0, y0), log2CbSize |
| pcm\_flag[ ][ ] | FL | cMax = 1 |
| prev\_intra\_luma\_pred\_flag[ ][ ] | FL | cMax = 1 |
| mpm\_idx[ ][ ] | TR | cMax = 2, cRiceParam = 0 |
| rem\_intra\_luma\_pred\_mode[ ][ ] | FL | cMax = 31 |
| intra\_chroma\_pred\_mode[ ][ ] | 9.3.3.6 | - |
| rqt\_root\_cbf | FL | cMax = 1 |
| prediction\_unit( ) | merge\_flag[ ][ ] | FL | cMax = 1 |
| merge\_idx[ ][ ] | TR | cMax = MaxNumMergeCand − 1, cRiceParam = 0 |
| inter\_pred\_idc[ x0 ][ y0 ] | 9.3.3.7 | nPbW, nPbH |
| ref\_idx\_l0[ ][ ] | TR | cMax = num\_ref\_idx\_l0\_active\_minus1, cRiceParam = 0 |
| mvp\_l0\_flag[ ][ ] | FL | cMax = 1 |
| ref\_idx\_l1[ ][ ] | TR | cMax = num\_ref\_idx\_l1\_active\_minus1, cRiceParam = 0 |
| mvp\_l1\_flag[ ][ ] | FL | cMax = 1 |
| transform\_tree( ) | split\_transform\_flag[ ][ ][ ] | FL | cMax = 1 |
| cbf\_luma[ ][ ][ ] | FL | cMax = 1 |
| cbf\_cb[ ][ ][ ] | FL | cMax = 1 |
| cbf\_cr[ ][ ][ ] | FL | cMax = 1 |
| mvd\_coding( ) | abs\_mvd\_greater0\_flag[ ] | FL | cMax = 1 |
| abs\_mvd\_greater1\_flag[ ] | FL | cMax = 1 |
| abs\_mvd\_minus2[ ] | EG1 | - |
| mvd\_sign\_flag[ ] | FL | cMax = 1 |
| transform\_unit( ) | cu\_qp\_delta\_abs | 9.3.3.8 | - |
| cu\_qp\_delta\_sign\_flag | FL | cMax = 1 |
| inter\_rdpcm\_flag[ ][ ][ ] | FL | cMax = 1 |
| inter\_rdpcm\_dir\_flag[ ][ ][ ] | FL | cMax = 1 |
| residual\_coding( ) | transform\_skip\_flag[ ][ ][ ] | FL | cMax = 1 |
| last\_sig\_coeff\_x\_prefix | TR | cMax = ( log2TrafoSize << 1 ) − 1, cRiceParam = 0 |
| last\_sig\_coeff\_y\_prefix | TR | cMax = ( log2TrafoSize << 1 ) − 1, cRiceParam = 0 |
| last\_sig\_coeff\_x\_suffix | FL | cMax = ( 1  <<  ( ( last\_sig\_coeff\_x\_prefix  >>  1 ) − 1 ) − 1 ) |
| last\_sig\_coeff\_y\_suffix | FL | cMax = ( 1  <<  ( ( last\_sig\_coeff\_y\_prefix  >>  1 ) − 1 ) − 1 ) |
| coded\_sub\_block\_flag[ ][ ] | FL | cMax = 1 |
| sig\_coeff\_flag[ ][ ] | FL | cMax = 1 |
| coeff\_abs\_level\_greater1\_flag[ ] | FL | cMax = 1 |
| coeff\_abs\_level\_greater2\_flag[ ] | FL | cMax = 1 |
| coeff\_abs\_level\_remaining[ ] | 9.3.3.9 | current sub-block scan index i, baseLevel |
| coeff\_sign\_flag[ ] | FL | cMax = 1 |

**8 Changes required to the derivation of context index variables and bypass**

The following rows are added to Table 9‑37 for the inter\_rdpcm\_flag and inter\_rdpcm\_dir\_flag flags.

| **Table 9‑37 – Assignment of ctxInc to syntax elements with context coded bins** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Syntax element** | **binIdx** | | | | | |
| **0** | **1** | **2** | **3** | **4** | **>= 5** |
| end\_of\_slice\_segment\_flag | terminate | na | na | na | na | na |
| end\_of\_sub\_stream\_one\_bit | terminate | na | na | na | na | na |
| sao\_merge\_left\_flag | 0 | na | na | na | na | na |
| sao\_merge\_up\_flag | 0 | na | na | na | na | na |
| sao\_type\_idx\_luma | 0 | bypass | na | na | na | na |
| sao\_type\_idx\_chroma | 0 | bypass | na | na | na | na |
| sao\_offset\_abs[ ][ ][ ][ ] | bypass | bypass | bypass | bypass | bypass | bypass |
| sao\_offset\_sign[ ][ ][ ][ ] | bypass | na | na | na | na | na |
| sao\_band\_position[ ][ ][ ] | bypass | bypass | bypass | bypass | bypass | bypass |
| sao\_eo\_class\_luma | bypass | bypass | bypass | na | na | na |
| sao\_eo\_class\_chroma | bypass | bypass | bypass | na | na | na |
| split\_cu\_flag[ ][ ] | 0,1,2 (subclause 9.3.4.2.2) | na | na | na | na | na |
| cu\_transquant\_bypass\_flag | 0 | na | na | na | na | na |
| cu\_skip\_flag | 0,1,2 (subclause 9.3.4.2.2) | na | na | na | na | na |
| pred\_mode\_flag | 0 | na | na | na | na | na |
| part\_mode log2CbSize = = MinCbLog2SizeY | 0 | 1 | 2 | bypass | na | na |
| part\_mode log2CbSize > MinCbLog2SizeY | 0 | 1 | 3 | bypass | na | na |
| pcm\_flag[ ][ ] | terminate | na | na | na | na | na |
| prev\_intra\_luma\_pred\_flag[ ][ ] | 0 | na | na | na | na | na |
| mpm\_idx[ ][ ] | bypass | bypass | na | na | na | na |
| rem\_intra\_luma\_pred\_mode[ ][ ] | bypass | bypass | bypass | bypass | bypass | bypass |
| intra\_chroma\_pred\_mode[ ][ ] | 0 | bypass | bypass | na | na | na |
| rqt\_root\_cbf | 0 | na | na | na | na | na |
| merge\_flag[ ][ ] | 0 | na | na | na | na | na |
| merge\_idx[ ][ ] | 0 | bypass | bypass | bypass | na | na |
| inter\_pred\_idc[ x0 ][ y0 ] | ( nPbW + nPbH ) != 12 ? CtDepth[ x0 ][ y0 ] : 4 | 4 | na | na | na | na |
| ref\_idx\_l0[ ][ ] | 0 | 1 | bypass | bypass | bypass | bypass |
| ref\_idx\_l1[ ][ ] | 0 | 1 | bypass | bypass | bypass | bypass |
| mvp\_l0\_flag[ ][ ] | 0 | na | na | na | na | na |
| mvp\_l1\_flag[ ][ ] | 0 | na | na | na | na | na |
| split\_transform\_flag[ ][ ][ ] | 5 − log2TrafoSize | na | na | na | na | na |
| cbf\_cb[ ][ ][ ] | trafoDepth | na | na | na | na | na |
| cbf\_cr[ ][ ][ ] | trafoDepth | na | na | na | na | na |
| cbf\_luma[ ][ ][ ] | trafoDepth = = 0 ? 1 : 0 | na | na | na | na | na |
| abs\_mvd\_greater0\_flag[ ] | 0 | na | na | na | na | na |
| abs\_mvd\_greater1\_flag[ ] | 0 | na | na | na | na | na |
| abs\_mvd\_minus2[ ] | bypass | bypass | bypass | bypass | bypass | bypass |
| mvd\_sign\_flag[ ] | bypass | na | na | na | na | na |
| cu\_qp\_delta\_abs | 0 | 1 | 1 | 1 | 1 | bypass |
| cu\_qp\_delta\_sign\_flag | bypass | na | na | na | na | na |
| transform\_skip\_flag[ ][ ][ ] | 0 | na | na | na | na | na |
| last\_sig\_coeff\_x\_prefix | 0..17 (subclause 9.3.4.2.3) | | | | | |
| last\_sig\_coeff\_y\_prefix | 0..17 (subclause 9.3.4.2.3) | | | | | |
| last\_sig\_coeff\_x\_suffix | bypass | bypass | bypass | bypass | bypass | bypass |
| last\_sig\_coeff\_y\_suffix | bypass | bypass | bypass | bypass | bypass | bypass |
| coded\_sub\_block\_flag[ ][ ] | 0..3 (subclause 9.3.4.2.4) | na | na | na | na | na |
| sig\_coeff\_flag[ ][ ] | 0..41 (subclause 9.3.4.2.5) | na | na | na | na | na |
| coeff\_abs\_level\_greater1\_flag[ ] | 0..23 (subclause 9.3.4.2.6) | na | na | na | na | na |
| coeff\_abs\_level\_greater2\_flag[ ] | 0..5 (subclause 9.3.4.2.7) | na | na | na | na | na |
| coeff\_abs\_level\_remaining[ ] | bypass | bypass | bypass | bypass | bypass | bypass |
| coeff\_sign\_flag[ ] | bypass | na | na | na | na | na |
| inter\_rdpcm\_flag[ ][ ][ ] | 0 | na | na | na | na | na |
| inter\_rdpcm\_dir\_flag[ ][ ][ ] | 0 | na | na | na | na | na |