**7.3.4.12 Residual coding syntax**

|  |  |  |
| --- | --- | --- |
| residual\_coding( x0, y0, log2TbWidth, log2TbHeight, cIdx ) { | **Descriptor** | |
| if( transform\_skip\_enabled\_flag && ( cIdx ! = 0  | |  tu\_mts\_flag[ x0 ][ y0 ] = = 0 ) &&   ( log2TbWidth  <=  2 ) && ( log2TbHeight  <=  2 ) ) |  | |
| **transform\_skip\_flag**[ x0 ][ y0 ][ cIdx ] | ae(v) | |
| **last\_sig\_coeff\_x\_prefix** | ae(v) |
| **last\_sig\_coeff\_y\_prefix** | ae(v) |
| if( last\_sig\_coeff\_x\_prefix > 3 ) |  |
| **last\_sig\_coeff\_x\_suffix** | ae(v) |
| if( last\_sig\_coeff\_y\_prefix > 3 ) |  |
| **last\_sig\_coeff\_y\_suffix** | ae(v) |
| log2SbSize = ( Min( log2TbWidth, log2TbHeight ) < 2 ? 1 : 2 ) |  |
| numSbCoeff = 1 << ( log2SbSize << 1 ) |  |
| lastScanPos = numSbCoeff |  |
| lastSubBlock = ( 1  <<  ( log2TbWidth + log2TbHeight − 2 \* log2SbSize ) ) − 1 |  |
| do { |  |
| if( lastScanPos = = 0 ) { |  |
| lastScanPos = numSbCoeff |  |
| lastSubBlock− − |  |
| } |  |
| lastScanPos− − |  |
| xS = DiagScanOrder[ log2TbWidth − log2SbSize ][ log2TbHeight − log2SbSize ]  [ lastSubBlock ][ 0 ] |  |
| yS = DiagScanOrder[ log2TbWidth − log2SbSize ][ log2TbHeight − log2SbSize ]  [ lastSubBlock ][ 1 ] |  |
| xC = ( xS << log2SbSize ) +   DiagScanOrder[ log2SbSize ][ log2SbSize ][ lastScanPos ][ 0 ] |  |
| yC = ( yS << log2SbSize ) +   DiagScanOrder[ log2SbSize ][ log2SbSize ][ lastScanPos ][ 1 ] |  |
| } while( ( xC != LastSignificantCoeffX ) | | ( yC != LastSignificantCoeffY ) ) |  |
|  |  |
| QState = 0 |  |
| for( i = lastSubBlock; i >= 0; i− − ) { |  |
| startQStateSb = QState |  |
| xS = DiagScanOrder[ log2TbWidth − log2SbSize ][ log2TbHeight − log2SbSize ]  [ lastSubBlock ][ 0 ] |  |
| yS = DiagScanOrder[ log2TbWidth − log2SbSize ][ log2TbHeight − log2SbSize ]  [ lastSubBlock ][ 1 ] |  |
| inferSbDcSigCoeffFlag = 0 |  |
| if( ( i < lastSubBlock ) && ( i > 0 ) ) { |  |
| **coded\_sub\_block\_flag**[ xS ][ yS ] | ae(v) |
| inferSbDcSigCoeffFlag = 1 |  |
| } |  |
| firstSigScanPosSb = numSbCoeff |  |
| lastSigScanPosSb = −1 |  |
| remBinsPass1 = ( log2SbSize < 2 ? 6 : 28 ) |  |
| remBinsPass2 = ( log2SbSize < 2 ? 2 : 4 ) |  |
| firstPosMode0 = ( i = = lastSubBlock ? lastScanPos : numSbCoeff − 1 ) |  |
| firstPosMode1 = −1 |  |
| firstPosMode2 = −1 |  |
| for( n = firstPosMode0; n >= 0 && remBinsPass1 >= 3; n− − ) { |  |
| xC = ( xS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ] |  |
| yC = ( yS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ] |  |
| if( coded\_sub\_block\_flag[ xS ][ yS ] && ( n > 0 | | !inferSbDcSigCoeffFlag ) && ( xC != LastSignificantCoeffX | | yC != Last SignificantCoeffY ) ) { |  |
| **sig\_coeff\_flag**[ xC ][ yC ] | ae(v) |
| remBinsPass1− − |  |
| if( sig\_coeff\_flag[ xC ][ yC ] ) |  |
| inferSbDcSigCoeffFlag = 0 |  |
| } |  |
| if( sig\_coeff\_flag[ xC ][ yC ] ) { |  | |
|  |  | |
| **abs\_level\_gt1\_flag**[ n ] | ae(v) |
| remBinsPass1− − |  |
| if( abs\_level\_gt1\_flag[ n ] ) { |  |
| **par\_level\_flag**[ n ] | ae(v) |
| remBinsPass1− − |  |
| if( remBinsPass2 > 0 ) { |  |
| remBinsPass2− − |  |
| if( remBinsPass2 = = 0 ) |  |
| firstPosMode1 = n − 1 |  |
| } |  |
| } |  |
| if( lastSigScanPosSb = = −1 ) |  |
| lastSigScanPosSb = n |  |
| firstSigScanPosSb = n |  |
| } |  |
| AbsLevelPass1[ xC ][ yC ] =   sig\_coeff\_flag[ xC ][ yC ] + par\_level\_flag[ n ] + abs\_level\_gt1\_flag[ n ] |  |
| if( dep\_quant\_enabled\_flag ) |  |
| QState = QStateTransTable[ QState ][ AbsLevelPass1[ xC ][ yC ] & 1 ] |  |
| if( remBinsPass1 < 3 ) |  |
| firstPosMode2 = n − 1 |  |
| } |  |
| if( firstPosMode1 < firstPosMode2 ) |  |
| firstPosMode1 = firstPosMode2 |  |
| for( n = numSbCoeff − 1; n >= firstPosMode1; n− − ) |  |
| if( abs\_level\_gt1\_flag[ n ] ) |  |
| **abs\_level\_gt3\_flag**[ n ] | ae(v) |
| for( n = numSbCoeff − 1; n >= firstPosMode1; n− − ) { |  |
| xC = ( xS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ] |  |
| yC = ( yS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ] |  |
| if( abs\_level\_gt3\_flag[ n ] ) |  |
| **abs\_remainder**[ n ] | ae(v) |
| AbsLevel[ xC ][ yC ] = AbsLevelPass1[ xC ][ yC ] +  2 \* ( abs\_level\_gt3\_flag[ n ] + abs\_remainder[ n ] ) |  |
| } |  |
| for( n = firstPosMode1; n > firstPosMode2; n− − ) { |  |
| xC = ( xS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ] |  |
| yC = ( yS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ] |  |
| if( abs\_level\_gt1\_flag[ n ] ) |  |
| **abs\_remainder**[ n ] | ae(v) |
| AbsLevel[ xC ][ yC ] = AbsLevelPass1[ xC ][ yC ] + 2 \* abs\_remainder[ n ] |  |
| } |  |
| for( n = firstPosMode2; n >= 0; n− − ) { |  |
| xC = ( xS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ] |  |
| yC = ( yS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ] |  |
| **dec\_abs\_level**[ n ] | ae(v) |
| if(AbsLevel[ xC ][ yC ] > 0 ) |  |
| firstSigScanPosSb = n |  |
| if( dep\_quant\_enabled\_flag ) |  |
| QState = QStateTransTable[ QState ][ AbsLevel[ xC ][ yC ] & 1 ] |  |
| } |  |
| if( dep\_quant\_enabled\_flag | | !sign\_data\_hiding\_enabled\_flag ) |  |
| signHidden = 0 |  |
| else |  |
| signHidden = ( lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0 ) |  |
| for( n = numSbCoeff − 1; n >= 0; n− − ) { |  |
| xC = ( xS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ] |  |
| yC = ( yS << log2SbSize ) + DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ] |  |
| if( ( AbsLevel[ xC ][ yC ] > 0 ) &&   ( !signHidden | | ( n != firstSigScanPosSb ) ) ) |  |
| **coeff\_sign\_flag**[ n ] | ae(v) |
| } |  |
| if( dep\_quant\_enabled\_flag ) { |  |
| QState = startQStateSb |  |
| for( n = numSbCoeff − 1; n >= 0; n− − ) { |  |
| xC = ( xS << log2SbSize ) +   DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ] |  |
| yC = ( yS << log2SbSize ) +   DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ] |  |
| if( AbsLevel[ xC ][ yC ] > 0 ) |  |
| TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =  ( 2 \* AbsLevel[ xC ][ yC ] − ( QState > 1 ? 1 : 0 ) ) \*  ( 1 − 2 \* coeff\_sign\_flag[ n ] ) |  |
| QState = QStateTransTable[ QState ][ par\_level\_flag[ n ] ] |  |
| } else { |  |
| sumAbsLevel = 0 |  |
| for( n = numSbCoeff − 1; n >= 0; n− − ) { |  |
| xC = ( xS << log2SbSize ) +   DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 0 ] |  |
| yC = ( yS << log2SbSize ) +   DiagScanOrder[ log2SbSize ][ log2SbSize ][ n ][ 1 ] |  |
| if( AbsLevel[ xC ][ yC ] > 0 ) { |  |
| TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =   AbsLevel[ xC ][ yC ] \* ( 1 − 2 \* coeff\_sign\_flag[ n ] ) |  |
| if( signHidden ) { |  |
| sumAbsLevel += AbsLevel[ xC ][ yC ] |  |
| if( ( n = = firstSigScanPosSb ) && ( sumAbsLevel % 2 ) = = 1 ) ) |  |
| TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =   −TransCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] |  |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |
| if(  tu\_mts\_flag[ x0 ][ y0 ]  &&  ( cIdx  = =  0 )  ) |  |
| **mts\_idx**[ x0 ][ y0 ][ cIdx ] | ae(v) |
| } |  | |

**9.5.3.2 Rice parameter derivation process for dec\_abs\_level[ ]**

Inputs to this process are the colour component index cIdx, the luma location ( x0, y0 ) specifying the top-left sample of the current transform block relative to the top-left sample of the current picture, the current coefficient scan location ( xC, yC ), the binary logarithm of the transform block width log2TbWidth, and the binary logarithm of the transform block height log2TbHeight.

Output of this process is the Rice parameter cRiceParam.

Given the array AbsLevel[ x ][ y ] for the transform block with component index cIdx and the top-left luma location ( x0, y0 ), the variable locSumAbs is derived as specified by the following pseudo code:

locSumAbs = 0  
if( xC < (1 << log2TbWidth) − 1 ) {  
 locSumAbs += AbsLevel[ xC + 1 ][ yC ]   
 if( xC < (1 << log2TbWidth) − 2 )  
 locSumAbs += AbsLevel[ xC + 2 ][ yC ]   
 if( yC < (1 << log2TbHeight) − 1 )  
 locSumAbs += AbsLevel[ xC + 1 ][ yC + 1 ] (9‑5)  
}  
if( yC < (1 << log2TbHeight) − 1 ) {  
 locSumAbs += AbsLevel[ xC ][ yC + 1 ]   
 if( yC < (1 << log2TbHeight) − 2 )  
 locSumAbs += AbsLevel [ xC ][ yC + 2 ]   
}   
if( locSumAbs > 31 )  
 locSumAbs = 31

The variable s is set equal to Max( 0, QState – 1 ).

Given the variables locSumAbs and s, the Rice parameter cRiceParam and the variable ZeroPos[ n ]0 are derived as specified in Table 9‑5.

**Table 9‑5 – Specification of cRiceParam and ZeroPos[ n ] based on locSumAbs and s**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **s** | **locSumAbs** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
|  | cRiceParam | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 |
| **0** | ZeroPos[ n ] | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 4 | 4 | 4 | 4 |
| **1** | ZeroPos[ n ] | 1 | 1 | 1 | 1 | 2 | 3 | 4 | 4 | 4 | 6 | 6 | 6 | 8 | 8 | 8 | 8 |
| **2** | ZeroPos[ n ] | 1 | 1 | 2 | 2 | 2 | 3 | 4 | 4 | 4 | 6 | 6 | 6 | 8 | 8 | 8 | 8 |
|  | **locSumAbs** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** |
|  | cRiceParam | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 |
| **0** | ZeroPos[ n ] | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 8 | 8 | 8 | 8 | 8 | 16 | 16 | 16 | 16 |
| **1** | ZeroPos[ n ] | 4 | 4 | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 16 | 16 | 16 | 16 | 16 | 16 |
| **2** | ZeroPos[ n ] | 8 | 8 | 12 | 12 | 12 | 12 | 12 | 12 | 12 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |