### 7.3.10 Residual coding syntax

|  |  |
| --- | --- |
| residual\_coding( x0, y0, log2TrafoWidth, log2TrafoHeight, scanIdx, cIdx ) { | Descriptor |
| **last\_significant\_coeff\_x\_prefix** | ae(v) |
| **last\_significant\_coeff\_y\_prefix** | ae(v) |
| if( last\_significant\_coeff\_x\_prefix > 3 ) |  |
| **last\_significant\_coeff\_x\_suffix** | ae(v) |
| if( last\_significant\_coeff\_y\_prefix > 3 ) |  |
| **last\_significant\_coeff\_y\_suffix** | ae(v) |
| numCoeff = 0 |  |
| numGreater1=0 |  |
| do { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ numCoeff ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ numCoeff ][ 1 ] |  |
| numCoeff++ |  |
| } while( ( xC != LastSignificantCoeffX ) | | ( yC != LastSignificantCoeffY ) ) |  |
| numLastSubset = (numCoeff − 1) >> 4 |  |
| for( i = numLastSubset; i >= 0; i− − ) { |  |
| offset = i << 4 |  |
| if( scanIdx = = 1 && log2TrafoWidth = = 3 && log2TrafoHeight = = 3 ) { |  |
| xCG = 0 |  |
| yCG = i |  |
| } else if( scanIdx = = 2 && log2TrafoWidth = = 3 && log2TrafoHeight = = 3 ) { |  |
| xCG = i |  |
| yCG = 0 |  |
| } else { |  |
| xCG = ScanOrder[ log2TrafoWidth − 2 ][ log2TrafoHeight − 2 ][ scanIdx ][ i ][ 0 ] |  |
| yCG = ScanOrder[ log2TrafoWidth − 2 ][ log2TrafoHeight − 2 ][ scanIdx ][ i ][ 1 ] |  |
| } |  |
| implicitNonZeroCoeff = 0 |  |
| if( (i < numLastSubset) && (i > 0) ) { |  |
| **significant\_coeff\_group\_flag**[ xCG ][ yCG ] | ae(v) |
| implicitNonZeroCoeff = 1 |  |
| } |  |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( (n + offset) < (numCoeff − 1) && significant\_coeff\_group\_flag[ xCG ][ yCG ] &&   ( n > 0 | | implicitNonZeroCoeff = = 0 ) ) { |  |
| **significant\_coeff\_flag**[ xC ][ yC ] | ae(v) |
| if( significant\_coeff\_flag[ xC ][ yC ] = = 1 ) |  |
| implicitNonZeroCoeff = 0 |  |
| } |  |
| } |  |
| firstNZPosInCG = 16 |  |
| lastNZPosInCG = −1 |  |
| numSigCoeff = 0 |  |
| firstGreater1CoeffIdx = −1 |  |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( significant\_coeff\_flag[ xC ][ yC ] ) { |  |
| if( numSigCoeff < 8 ) { |  |
| **coeff\_abs\_level\_greater1\_flag[** n **]** | ae(v) |
| numSigCoeff++ |  |
| if( coeff\_abs\_level\_greater1\_flag[ n ] && firstGreater1CoeffIdx = = −1 ) |  |
| firstGreater1CoeffIdx = n |  |
| } |  |
| if( lastNZPosInCG = = −1) |  |
| lastNZPosInCG = n |  |
| firstNZPosInCG = n |  |
| } |  |
| } |  |
| signHidden = ( lastNZPosInCG – firstNZPosInCG >= sign\_hiding\_threshold) ? 1 : 0 |  |
| if( firstGreater1CoeffIdx != −1 ) |  |
| **coeff\_abs\_level\_greater2\_flag[** firstGreater1CoeffIdx**]** | ae(v) |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( significant\_coeff\_flag[ xC ][ yC ] &&  (!sign\_data\_hiding\_flag | | !signHidden | | n != firstNZPosInCG) ) |  |
| **coeff\_sign\_flag[** n **]** | ae(v) |
| } |  |
| numSigCoeff = 0 |  |
| sumAbs = 0 |  |
| numGreater1 = numGreater1 >> 1 |  |
| for( n = 15; n >= 0; n− − ) { |  |
| xC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 0 ] |  |
| yC = ScanOrder[ log2TrafoWidth ][ log2TrafoHeight ][ scanIdx ][ n + offset ][ 1 ] |  |
| if( significant\_coeff\_flag[ xC ][ yC ] ) { |  |
| baseLevel = 1 + coeff\_abs\_level\_greater1\_flag[ n ] + coeff\_abs\_level\_greater2\_flag[ n ] |  |
| if( baseLevel = = ( ( numSigCoeff < 8 ) ? ( (n = = firstGreater1CoeffIdx) ? 3 : 2 ) : 1 ) ) |  |
| **coeff\_abs\_level\_remaining[** n **]** | ae(v) |
| transCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] =   ( coeff\_abs\_level\_remaining[ n ] + baseLevel ) \* ( 1 − 2 \* coeff\_sign\_flag[ n ] ) |  |
| if ( ( coeff\_abs\_level\_remaining[ n ] + baseLevel ) > 1 )  numGreater1 +=1 |  |
| if( sign\_data\_hiding\_flag && signHidden ) { |  |
| sumAbs += ( coeff\_abs\_level\_remaining[ n ] + baseLevel ) |  |
| if( n = = firstNZPosInCG && (sumAbs%2 = = 1) ) |  |
| transCoeffLevel[x0][y0][cIdx][xC][yC] = −  transCoeffLevel[x0][y0][cIdx][xC][yC] |  |
| } |  |
| numSigCoeff++ |  |
| } else |  |
| transCoeffLevel[ x0 ][ y0 ][ cIdx ][ xC ][ yC ] = 0 |  |
| } |  |
| } |  |
| } |  |

##### 9.2.3.1.5 Derivation process of ctxIdxInc for the syntax element coeff\_abs\_level\_greater1\_flag

Inputs to this process are the colour component index cIdx, the 16 coefficient subset index i and the current coefficient scan index n within the current subset.

Output of this process is ctxIdxInc.

The variable ctxSet specifies the current context set and for its derivation the following applies.

* If n is equal to 15 or all previous syntax elements coeff\_abs\_level\_greater1\_flag[ pos ] with pos greater than n are derived to be equal to 0 instead of being explicitly parsed, the following applies.
* The variable ctxSet is initialized as follows.
* If the current subset index i is equal to 0 or cIdx is greater than 0, the following applies.

ctxSet = 0 (9‑31)

* Otherwise (i is greater than 0 and cIdx is equal to 0), the following applies.

ctxSet = 2 (9‑32)

* When the subset i is not the first one to be processed in this subclause, the following applies.
* The variable numGreater1 is set equal to the variable numGreater1 that has been derived in subclause 7.3.10.
* When numGreater1 is greater than 0, ctxSet is incremented by one as follows.

ctxSet = ctxSet + 1 (9‑33)

* The variable greater1Ctx is set equal to 1.
* Otherwise (coeff\_abs\_level\_greater1\_flag[ n ] is not the first to be parsed within the current subset i),.for the derivation of ctxSet and greater1Ctx the following applies.
* The variable ctxSet is set equal to the variable ctxSet that has been derived during the last invocation of this subclause.
* The variable greater1Ctx is set equal to the variable greater1Ctx that has been derived during the last invocation of this subclause.
* When greater1Ctx is greater than 0, the variable lastGreater1Flag is set equal to the syntax element coeff\_abs\_level\_greater1\_flag that has been used during the last invocation of this subclause and greater1Ctx is modifed as follows.
* If lastGreater1Flag is equal to 1, greater1Ctx is set equal to 0.
* Otherwise (lastGreater1Flag is equal to 0), greater1Ctx is incremented by 1.

The context index increment ctxIdxInc is derived using the current context set ctxSet and the current context greater1Ctx as follows.

ctxIdxInc = ( ctxSet \* 4 ) + Min( 3, greater1Ctx ) (9‑34)

When cIdx is greater than 0, ctxIdxInc is modified as follows.

ctxIdxInc = ctxIdxInc + 16 (9‑35)

##### 9.2.3.1.6 Derivation process of ctxIdxInc for the syntax element coeff\_abs\_level\_greater2\_flag

Inputs to this process are the colour component index cIdx, the 16 coeffcient subset index i and the current coefficient scan index n within the current subset.

Output of this process is ctxIdxInc.

The variable ctxSet specifies the current context set and is set to the variable ctxSet that has been derived in subclause 9.2.3.1.5 for the same subset.

The context index increment ctxIdxInc is set to the variable ctxSet as follows.

ctxIdxInc = ctxSet (9‑38)

When cIdx is greater than 0, ctxIdxInc is modified as follows.

ctxIdxInc = ctxIdxInc + 4 (9‑39)