# Syntax changes

|  |  |
| --- | --- |
| pred\_weight\_table( ) { | Descriptor |
| **luma\_log2\_weight\_denom** | ue(v) |
| if( chroma\_format\_idc != 0 ) |  |
| **delta\_chroma\_log2\_weight\_denom** | se(v) |
|  |  |
| for( i = 0; i <= num\_ref\_idx\_l0\_active\_minus1; i++ ) { |  |
| **luma\_weight\_l0\_flag** [ i ] | u(1) |
| if( luma\_weight\_l0\_flag [ i ] ) { |  |
| **delta\_luma\_weight\_l0[** i **]** | se(v) |
| **delta\_luma\_offset\_l0[** i **]** | se(v) |
| } |  |
| if( chroma\_format\_idc != 0 ) { |  |
| **chroma\_weight\_l0\_flag** [ i ] | u(1) |
| if( chroma\_weight\_l0\_flag [ i ] ) |  |
| for( j =0; j < 2; j++ ) { |  |
| **delta\_chroma\_weight\_l0[** i **][** j **]** | se(v) |
| **delta\_chroma\_offset\_l0[** i **][** j **]** | se(v) |
| } |  |
| } |  |
| } |  |
|  |  |
| if( slice\_type = = B ) { |  |
| if( ref\_pic\_list\_combination\_flag = = 0 ) |  |
| **weights\_l1\_present\_flag** | u(1) |
| if( weights\_l1\_present\_flag ) { |  |
| **delta\_params\_present\_flag [i]** | u(1) |
| if (delta\_params\_present\_flag [i]) { |  |
| for( i = 0; i <= num\_ref\_idx\_l1\_active\_minus1; i++ ) { |  |
| **luma\_weight\_l1\_flag**[ i ] | u(1) |
| if( luma\_weight\_l1\_flag [ i ]) { |  |
| **delta\_luma\_weight\_l1[** i **]** | se(v) |
| **delta\_luma\_offset\_l1[** i **]** | se(v) |
| } |  |
| if( chroma\_format\_idc != 0 ) { |  |
| **chroma\_weight\_l1\_flag**[ i ] | u(1) |
| if( chroma\_weight\_l1\_flag [ i ]) |  |
| for( j = 0; j < 2; j++ ) { |  |
| **delta\_chroma\_weight\_l1[** i **][** j **]** | se(v) |
| **delta\_chroma\_offset\_l1[** i **][** j **]** | se(v) |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| } |  |
| } |  |

# Semantics changes

#### 7.4.3.8 Weighted prediction parameters semantics

**luma\_log2\_weight\_denom** is the base 2 logarithm of the denominator for all luma weighting factors. The value of luma\_log2\_weight\_denom shall be in the range of 0 to 7, inclusive.

**delta\_chroma\_log2\_weight\_denom** is the difference of the base 2 logarithm of the denominator for all chroma weighting factors.

The variable ChromaLog2WeightDenom is specified by luma\_log2\_weight\_denom + delta\_chroma\_log2\_weight\_denom and it shall be in the range of 0 to 7, inclusive.

**luma\_weight\_l0\_flag** [ i ] equal to 1 specifies that weighting factors for the luma component of list 0 prediction using RefPicList0[ i ] are present. luma\_weight\_l0\_flag[i] equal to 0 specifies that these weighting factors are not present.

**delta\_luma\_weight\_l0[** i **]** is the difference of the weighting factor applied to the luma prediction value for list 0 prediction using RefPicList0[ i ].

The variable LumaWeightL0[ i ] is specified by invoking the decoding process in 8.3.7. The value of LumaWeightL0[ i ] shall be in the range of −128 to 127, inclusive. **delta\_luma\_offset\_l0[** i **]** is the difference of the additive offset applied to the luma prediction value for list 0 prediction using RefPicList0[ i ]. The variable LumaOffsetL0[ i ] is specified by invoking the decoding process in 8.3.7. The value of LumaOffsetL0[ i ] shall be in the range of −128 to 127, inclusive.

**chroma\_weight\_l0\_flag** [ i ] equal to 1 specifies that weighting factors for the chroma prediction values of list 0 prediction using RefPicList0[ i ] are present. chroma\_weight\_l0\_flag[i] equal to 0 specifies that these weighting factors are not present.

**delta\_chroma\_weight\_l0[** i **][** j **]** is the difference of the weighting factor applied to the chroma prediction values for list 0 prediction using RefPicList0[ i ] with j equal to 0 for Cb and j equal to 1 for Cr.

The variable ChromaWeightL0[ i ][ j ] is specified by invoking the decoding process in 8.3.7. The value of ChromaWeightL0[ i ][ j ] shall be in the range of −128 to 127, inclusive. **delta\_chroma\_offset\_l0[** i **][** j **]** is the difference of the additive offset applied to the chroma prediction values for list 0 prediction using RefPicList0[ i ] with j equal to 0 for Cb and j equal to 1 for Cr.

The variable ChromaOffsetL0[ i ][ j ] is specified by invoking the process in 8.3.7.

The variable ChromaOffsetL0[ i ][ j ] shall be in the range of −127 to 128, inclusive.

**weights\_l1\_present\_flag** equal to 1 specifies that syntax elements used to derive the weighting factors LumaWeightL1[ i ] and ChromaWeightL1 [ i ][ j ] and the additive offsets LumaOffsetL1[ i ] and ChromaOffsetL1 [ i ][ j ] for the luma and chroma prediction values of list 1 prediction are present. When weights\_l1\_present\_flag is equal to 0, syntax elements used to derive the weighting factors and the additive offsets for luma and chroma prediction values of list 1 prediction are not present. When weights\_l1\_present\_flag is not present, it is inferred to be 1.

**delta\_params\_present\_flag[i]** equal to 1 specifies that the syntax elements luma\_weight\_l1\_flag, delta\_luma\_weight\_l1, delta\_luma\_offset\_l1, chroma\_weight\_l1\_flag, delta\_chroma\_weight\_l1, delta\_chroma\_offset\_l1 are present. When delta\_params\_present\_flag is equal to 0, the syntax elements luma\_weight\_l1\_flag, delta\_luma\_weight\_l1, delta\_luma\_offset\_l1, chroma\_weight\_l1\_flag, delta\_chroma\_weight\_l1, delta\_chroma\_offset\_l1 are not present.

**luma\_weight\_l1\_flag, delta\_luma\_weight\_l1**, delta\_**luma\_offset\_l1**, **chroma\_weight\_l1\_flag**, **delta\_chroma\_weight\_l1**, **delta\_chroma\_offset\_l1** have the same semantics as luma\_weight\_l0\_flag, delta\_luma\_weight\_l0, delta\_luma\_offset\_l0, chroma\_weight\_l0\_flag, delta\_chroma\_weight\_l0, delta\_chroma\_offset\_l0, respectively, with l0, list 0, and List0 replaced by l1, list 1, and List1, respectively.

The variables LumaWeightL1[ i ], LumaOffsetL1[ i ], ChromaWeightL1[ i ][ j ] and ChromaWeightL1[ i ][ j ] are specified by invoking the decoding process in 8.3.7. The variables LumaWeightL1[ i ], LumaOffsetL1[ i ], ChromaWeightL1[ i ][ j ] and ChromaWeightL1[ i ][ j ] shall be in the range of -127 to 128, inclusive.

# Decoding process changes

It is proposed that 8.3 Slice decoding process be appended with the following subclause 8.3.7.

### 8.3.7 Decoding process for weighted prediction parameters

This process is invoked at the beginning of the decoding process for each P slice when weighted\_pred\_flag is equal to 1 or at the beginning of the decoding process for each B slice when weighted\_bipred\_idc is equal to 1.

Inputs to this process are:

– luma\_log2\_weight\_denom and ChromaLog2WeightDenom,

– for each entry in List 0, luma\_weight\_l0\_flag, delta\_luma\_weight\_l0, delta\_luma\_offset\_l0, chroma\_weight\_l0\_flag, delta\_chroma\_weight\_l0, and delta\_chroma\_offset\_l0,

– if this is a B slice, for each entry in List 1, luma\_weight\_l1\_flag, delta\_luma\_weight\_l1, delta\_luma\_offset\_l1, chroma\_weight\_l1\_flag, delta\_chroma\_weight\_l1, and delta\_chroma\_offset\_l1.

Outputs of this process are arrays of the weighting factors and the additive offsets for luma and chroma weighted sample prediction:

– for each entry in List 0, LumaWeightL0, LumaOffsetL0, ChromaWeightL0, and ChromaOffsetL0,

– if this is a B slice, for each entry in List 1, LumaWeightL1, LumaOffsetL1, ChromaWeightL1, and ChromaOffsetL1.

The following ordered steps apply:

The array RefPicList0ToRPS is constructed as follows:

for( cIdx = 0; cIdx ≤ num\_ref\_idx\_l0\_active\_minus1; cIdx++)  
 RefPicList0ToRPS[ cIdx ] = ref\_pic\_list\_modification\_flag\_l0 ? list\_entry\_l0[ cIdx ] : cIdx

If the slice is a B slice, the array RefPicList1ToRPS is constructed as follows:

for( cIdx = 0; cIdx ≤ num\_ref\_idx\_l1\_active\_minus1; cIdx++ ) {  
 tempIdx = ref\_pic\_list\_modification\_flag\_l1 ? list\_entry\_l1[ cIdx ] : cIdx   
 RefPicList1ToRPS[ cIdx ] = tempIdx < NumPocStCurrAfter ? tempIdx +NumPocStCurrBefore :   
 tempIdx < NumPocStCurrBefore+NumPocStCurrAfter ? tempIdx – NumPocStCurrAfter : tempIdx   
}

The arrays LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, ChromaOffsetPredInit are initialized as follows:

Let cIdx be the index into the prediction arrays LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, and ChromaOffsetPredInit.

The following process is repeated for all cIdx between 0 and NumPocTotalCurr-1:

LumaWeightPred [ cIdx ] = (1 << luma\_log2\_weight\_denom ),   
 ChromaWeightPred [ cIdx ][ 0 ] = (1 << ChromaLog2WeightDenom ),

ChromaWeightPred [ cIdx ][ 1 ] = (1 << ChromaLog2WeightDenom ),

LumaOffsetPred [ cIdx ] = 0,   
 ChromaOffsetPred [ cIdx ][ 0 ] = 0,

ChromaOffsetPred [ cIdx ][ 1 ] = 0,

ChromaOffsetPredInit [ cIdx ] = 0

The arrays LumaWeightL0, LumaOffsetL0, ChromaWeightL0, ChromaOffsetL0 are constructed by invoking the process in 8.3.7.1 with luma\_weight\_l0\_flag, delta\_luma\_weight\_l0, delta\_luma\_offset\_l0, chroma\_weight\_l0\_flag, delta\_chroma\_weight\_l0, delta\_chroma\_offset\_l0, LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, ChromaOffsetPredInit, and RefPicList0ToRPS as inputs and LumaWeightL0, LumaOffsetL0, ChromaWeightL0, ChromaOffsetL0, LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, and ChromaOffsetPredInit as outputs.

When the slice is a B slice,

If weights\_l1\_present\_flag is equal to 0, the arrays LumaWeightL1, LumaOffsetL1, ChromaWeightL1, ChromaOffsetL1 are constructed as follows:

for( cIdx = 0; cIdx ≤ num\_ref\_idx\_l1\_active\_minus1; cIdx++ ) {  
 LumaWeightL1 [ cIdx ] = LumaWeightL0[ cIdx ]   
 LumaOffsetL1 [ cIdx ] = LumaOffsetL0[ cIdx ]

ChromaWeightL1 [ cIdx ][ 0 ] = ChromaWeightL0[ cIdx ][ 0 ]

ChromaWeightL1 [ cIdx ][ 1 ] = ChromaWeightL0[ cIdx ][ 1 ]

ChromaOffsetL1 [ cIdx ][ 0 ] = ChromaOffsetL0[ cIdx ][ 0 ]

ChromaOffsetL1 [ cIdx ][ 1 ] = ChromaOffsetL0[ cIdx ][ 1 ]

}

Otherwise (if weights\_l1\_present\_flag is equal to 1), the arrays LumaWeightL1, LumaOffsetL1, ChromaWeightL1, ChromaOffsetL1 are constructed by invoking the process in 8.3.7.1 with luma\_weight\_l1\_flag, delta\_luma\_weight\_l1, delta\_luma\_offset\_l1, chroma\_weight\_l1\_flag, delta\_chroma\_weight\_l1, delta\_chroma\_offset\_l1, LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, ChromaOffsetPredInit, and RefPicList1ToRPS as inputs and LumaWeightL1, LumaOffsetL1, ChromaWeightL1, ChromaOffsetL1, LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, and ChromaOffsetPredInit as outputs.

###### 8.3.7.1 Weighted parameter decoding process for List X

Inputs to this process are:

– luma\_log2\_weight\_denom and ChromaLog2WeightDenom,

– for each entry in List X (with X being replaced by 0 or 1), luma\_weight\_lx\_flag, delta\_luma\_weight\_lx, delta\_luma\_offset\_lx, chroma\_weight\_lx\_flag, delta\_chroma\_weight\_lx, and delta\_chroma\_offset\_lx,

– for each entry in List X if X is equal to 1, delta\_params\_present\_flag,

– the array RefPicListXToRPS (with X being replaced by 0 or 1),

– the arrays LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, ChromaOffsetPredInit.

Outputs of this process are:

– for each entry in List X (with X being replaced by 0 or 1), LumaWeightLX, LumaOffsetLX, ChromaWeightLX, and ChromaOffsetLX,

– the updated arrays LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred, ChromaOffsetPredInit.

The following process is repeated for each cIdx in List X in the range of 0 and num\_ref\_idx\_lx\_active\_minus1 (with X being replaced by 0 or 1), inclusively:

–Let tempIdx be equal to RefPicListXToRPS [ cIdx ]

–If delta\_params\_present\_flag[i] is 0,

LumaWeightLX[ cIdx ] = LumaWeightPred [ tempIdx ]   
 LumaOffsetLX[ cIdx ] = LumaOffsetPred [ tempIdx ]

ChromaWeightLX[ cIdx ][0] = ChromaWeightPred[tempIdx][0]   
 ChromaWeightLX[ cIdx ][1] = ChromaWeightPred[tempIdx][1]

ChromaOffsetLX[ cIdx ][0] = ChromaOffsetPred[tempIdx][0]   
 ChromaOffsetLX[ cIdx ][1] = ChromaOffsetPred[tempIdx][1]

– Otherwise (if delta\_params\_present\_flag is 1),

–If luma\_weight\_lx\_flag[ cIdx ] is 1,

LumaWeightLX[ cIdx ] = LumaWeightPred [ tempIdx ] + delta\_luma\_weight\_lx[ cIdx ]   
 LumaOffsetLX[ cIdx ] = LumaOffsetPred [ tempIdx ] + delta\_luma\_offset\_lx[ cIdx ]

– Otherwise (if luma\_weight\_lx\_flag[ cIdx] is 0),

LumaWeightLX[ cIdx ] = (1 << luma\_log2\_weight\_denom)   
 LumaOffsetLX[ cIdx ] = 0

–If chroma\_weight\_lx\_flag[ cIdx ] is 1,

ChromaWeightLX[ cIdx ][0] = ChromaWeightPred[tempIdx][0] + delta\_chroma\_weight\_lx[ cIdx ][0]   
 ChromaWeightLX[ cIdx ][1] = ChromaWeightPred[tempIdx][1] + delta\_chroma\_weight\_lx[ cIdx ][1]

– If ChromaOffsetPredInit[ tempIdx ] is equal to 0, ChromaOffsetPred[ tempIdx ][ 0 ], ChromaOffsetPred[ tempIdx][ 1 ], and ChromaOffsetPredInt[tempIdx] are set as follows:

ChromaOffsetPred[ tempIdx ] [ 0 ] = (1 << (BitDepthC – 1) ) –

( ChromaWeightLX[ cIdx ][ 0 ] << (BitDepthC – 1 – ChromaLog2WeightDenom ))

ChromaOffsetPred[ tempIdx ] [ 1 ] = (1 << (BitDepthC – 1) ) –

( ChromaWeightLX[ cIdx ][ 1 ] << (BitDepthC – 1 – ChromaLog2WeightDenom ))

ChromaOffsetPredInit[ tempIdx ] = 1

ChromaOffsetLX[ cIdx ][ 0 ] = ChromaOffsetPred [ tempIdx ][0] + delta\_chroma\_weight\_lx[ cIdx ][0]  
 ChromaOffsetLX[ cIdx ][ 1 ] = ChromaOffsetPred [ tempIdx ][1] + delta\_chroma\_weight\_lx[ cIdx ][1]

– Otherwise (if chroma\_weight\_lx\_flag[ cIdx] is 0),

ChromaWeightLX[ cIdx ][ 0 ] = (1 << ChromaLog2WeightDenom)   
 ChromaWeightLX[ cIdx ][ 1 ] = (1 << ChromaLog2WeightDenom)   
 ChromaOffsetLX[ cIdx ][ 0 ] = 0

ChromaOffsetLX[ cIdx ][ 1 ] = 0

– The arrays LumaWeightPred, LumaOffsetPred, ChromaWeightPred, ChromaOffsetPred are updated as follows:

LumaWeightPred [ tempIdx ] = LumaWeightLX[ cIdx ]  
LumaOffsetPred [ tempIdx ] = LumaOffsetLX[ cIdx ]  
ChromaWeightPred [ tempIdx ][ 0 ] = ChromaWeightLX[ cIdx ] [ 0 ]   
ChromaWeightPred [ tempIdx ][ 1 ] = ChromaWeightLX[ cIdx ] [ 1 ]   
ChromaOffsetPred [ tempIdx ][ 0 ] = ChromaOffsetLX[ cIdx ][ 0 ]   
ChromaOffsetPred [ tempIdx ][ 1 ] = ChromaOffsetLX[ cIdx ][ 1 ]

###### 8.5.2.2.3.2 Weighted sample prediction process

Inputs to this process are:

– the same as specified in subclause .

– Outputs of this process are:

– the same as specified in subclause .

The prediction sample predSamples[ x, y ] with x = 0..(nPSW)−1 and y = 0..(nPSH)−1, with H being replaced by Y for luma samples and by C for chroma samples are derived as follows:

– If the predFlagL0 is equal to 1 and predFlagL1 is equal to 0, the prediction samples are derived as follows:

if( logWDC >= 1 )   
 predSamples[ x, y ] = Clip1H( ( ( predSamplesL0 [ x, y ] \* w0C + 2logWDC − 1 ) >> logWDC ) + o0C ) (8‑215)  
else  
 predSamples[ x, y ] = Clip1H( predSamplesL0 [ x, y ] \* w0C + o0C ) (8‑216)

– Otherwise, if the predFlagL0 is equal to 0 and predFlagL1 is equal to 1, the final predicted sample values predSamples [ x, y ] are derived by

if( logWDC >= 1 )  
 predSamples[ x, y ] = Clip1H( ( ( predSamplesL1 [ x, y ] \* w1C + 2logWDC − 1 ) >> logWDC ) + o1C ) (8‑217)  
else  
 predSamples[ x, y ] = Clip1H( predSamplesL1 [ x, y ] \* w1C + o1C ) (8‑218)

– Otherwise (both predFlagL0 and predFlagL1 are equal to 1), if RefPicOrderCnt( currPic, refIdxL0, L0) is equal to RefPicOrderCnt( currPic, refIdxL1, L1) and mvL0 is equal to mvL1, the final predicted sample values predSamples [ x, y ] are derived by

predSamples[ x, y ] = Clip1H( ( predSamplesL0 [ x, y ] \* ( w0C + w1C ) +   
 ( ( o0C + o1C + 1 ) << logWDC ) ) >> ( logWDC + 1 ) ) (8‑219)

– Otherwise, the final predicted sample values predSamples[ x, y ] are derived by

predSamples[ x, y ] = Clip1H( ( predSamplesL0 [ x, y ] \* w0C + predSamplesL1 [ x, y ] \* w1C +   
 ( ( o0C + o1C + 1 ) << logWDC ) ) >> ( logWDC + 1 ) ) (8‑220)

Where the variables logWDC, o0C, o1C, and w0C, w1C are derived as follows.

– If weighted\_bipred\_idc is equal to 2 in B-slices, implicit mode weighted prediction is used as follows:

logWDc = 5+shift1 (8‑221)

o0C = 0 (8‑222)

o1C = 0 (8‑223)

The variable WeightScaleFactor is derived from the values currPoc, refIdxL0 and refIdxL1 as follows:

tb = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListL0( refIdxL0 ) ) ) (8‑224)

td = Clip3( −128, 127, PicOrderCnt( RefPicListL1( refIdxL1 ) )   
  – PicOrderCnt( RefPicListL0( refIdxL0 ) ) ) (8‑225)

tx = ( 16384 + ( Abs( td ) >> 1 ) ) / td (8‑226)

WeightScaleFactor = Clip3( −1024, 1023, ( tb \* tx + 32 ) >> 6 ) (8‑227)

The variables w0C and w1C are derived as follows.

– If PicOrderCnt( RefPicListL0( refIdxL0 ) ) is equal to PicOrderCnt( RefPicListL1( refIdxL1 ) ) or ( WeightScaleFactor >> 2 ) < −64 or ( WeightScaleFactor >> 2 ) > 128, the following applies.

w0C=32 (8‑228)

w1C=32 (8‑229)

– Otherwise;

w0C = 64 – (WeightScaleFactor >> 2) (8‑230)

w1C = WeightScaleFactor >> 2 (8‑231)

~~– Otherwise, if both weighted\_bipred\_idc and ref\_pic\_list\_combination\_flag are equal to 1 in B slice, combined explicit mode weighted prediction is used as follows:~~

~~– If C is equal to L for luma samples,~~

~~logWDc = luma\_log2\_weight\_denom+ shift1 (8‑232)~~

~~w~~~~0C~~ ~~= LumaWeightLC[ RefIdxLCToRefIdxLx[ refIdxL0 ] ] (8‑233)~~

~~w~~~~1C~~ ~~= LumaWeightLC[ RefIdxLCToRefIdxLx[ refIdxL1 ] ] (8‑234)~~

~~o~~~~0C~~ ~~= luma\_offset\_lc[ RefIdxLCToRefIdxLx[ refIdxL0 ] ] \* ( 1 << ( BitDepth~~~~Y~~ ~~− 8 ) ) (8‑235)~~

~~o~~~~1C~~ ~~= luma\_offset\_lc[ RefIdxLCToRefIdxLx[ refIdxL1 ] ] \* ( 1 << ( BitDepth~~~~Y~~ ~~− 8 ) ) (8‑236)~~

~~– Otherwise (C is equal to Cb or Cr for chroma samples, with iCbCr = 0 for Cb, iCbCr = 1 for Cr),~~

~~logWDc = ChromaLog2WeightDenom + shift1 (8‑237)~~

~~w~~~~0C~~ ~~= ChromaWeightLC[ RefIdxLCToRefIdxLx[ refIdxL0 ] ][ iCbCr ] (8‑238)~~

~~w~~~~1C~~ ~~= ChromaWeightLC[ RefIdxLCToRefIdxLx[ refIdxL1 ] ][ iCbCr ] (8‑239)~~

~~o~~~~0C~~ ~~= ChromaOffsetLC[ RefIdxLCToRefIdxLx[ refIdxL0 ] ][ iCbCr ] \* ( 1 << ( BitDepth~~~~C~~ ~~− 8 ) ) (8‑240)~~

~~o~~~~1C~~ ~~= ChromaOffsetLC[ RefIdxLCToRefIdxLx[ refIdxL1 ] ][ iCbCr ] \* ( 1 << ( BitDepth~~~~C~~ ~~− 8 ) ) (8‑241)~~

– Otherwise (if weighted\_pred\_flag is equal to 1 in P slice or if weight\_bipred\_idc is equal to 1 in B-slice) explicit mode weighted prediction is used as follows:

– If C is equal to L for luma samples,

logWDc = luma\_log2\_weight\_denom+ shift1 (8‑242)

w0C = LumaWeightL0[refIdxL0] (8‑243)

w1C = LumaWeightL1[refIdxL1] (8‑244)

o0C = LumaOffsetL0[refIdxL0] \* ( 1 << ( BitDepthY − 8 ) ) (8‑245)

o1C = LumaOffsetL1[refIdxL1] \* ( 1 << ( BitDepthY − 8 ) ) (8‑246)

– Otherwise (C is equal to Cb or Cr for chroma samples, with iCbCr = 0 for Cb, iCbCr = 1 for Cr),

logWDc = ChromaLog2WeightDenom + shift1 (8‑247)

w0C = ChromaWeightL0[refIdxL0][ iCbCr ] (8‑248)

w1C = ChromaWeightL1[refIdxL1][ iCbCr ] (8‑249)

o0C = ChromaOffsetL0[refIdxL0][ iCbCr ] \* ( 1 << ( BitDepthC − 8 ) ) (8‑250)

o1C = ChromaOffsetL1[refIdxL1][ iCbCr ] \* ( 1 << ( BitDepthC − 8 ) ) (8‑251)