### Derivation process for luma intra prediction mode

Input to this process is a luma location ( xB, yB ) specifying the top-left luma sample of the current block relative to the top‑left luma sample of the current picture.

[Ed. (YK): Should the output of this process be stated here? The output statement is missing for only two places where there are a statement of input(s).]

specifies the value for the intra prediction mode and the associated names.

Table 8‑1 – Specification of intra prediction mode and associated names

|  |  |
| --- | --- |
| **Intra prediction mode** | **Associated name** |
| 0 | INTRA\_PLANAR |
| 1 | INTRA\_DC |
| 2..34 | INTRA\_ANGULAR2..INTRA\_ANGULAR34 |
| 35 | Intra\_FromLuma (used only for chroma) |



##### General intra sample prediction

Inputs to this process are:

– a sample location ( xB, yB ) specifying the top-left sample of the current block relative to the top‑left sample of the current picture,

– a variable predModeIntra specifying the intra prediction mode,

– a variable nT specifying the transform block size,

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

Output of this process is the predicted samples predSamples[ x ][ y ], with x, y = 0..nT − 1.

The nT \* 4 + 1 neighbouring samples p[ x ][ y ] that are constructed samples prior to the deblocking filter process, with x = −1, y = −1..nT \* 2 − 1 and x = 0..nT \* 2 − 1, y = −1, are derived as follows:

– The neighbouring location (xN, yN ) is specified by

( xN, yN ) = ( xB + x, yB + y ) (8‑26)

* The current luma location ( xBY, yBY ) and the neighbouring luma location (xNY, yNY ) are derived as follows:

( xBY, yBY ) = ( cIdx  = =  0 ) ? ( xB, yB ) : ( xB \* SubWidthC, yB \* SubHeightC ) (8‑27)

( xNY, yNY ) = ( cIdx  = =  0 ) ? ( xN, yN ) : ( xN \* SubWidthC, yN \* SubHeightC ) (8‑28)

* The availability derivation process for a block in z-scan order as specified in subclause 6.4.1 is invoked with the current luma location ( xCurr, yCurr ) set equal to ( xBY, yBY ) and the neighbouring luma location ( xNY, yNY ) as inputs, and the output is assigned to availableN.

– Each sample p[ x ][ y ] is derived as follows:

* If one or more of the following conditions are true, the sample p[ x ][ y ] is marked as "not available for intra prediction":
  + The variable availableN is equal to FALSE.
  + CuPredMode[ xNY ][ yNY ] is not equal to MODE\_INTRA and constrained\_intra\_pred\_flag is equal to 1.
* Otherwise, the sample p[ x ][ y ] is marked as "available for intra prediction" and the sample at the location ( xN, yN ) is assigned to p[ x ][ y ].

When chroma\_pred\_from\_luma\_enabled\_flag is equal to 1, cIdx is equal to 1 or 2, and intraPredMode is equal to Intra\_FromLuma, the nT\*8+1 neighbouring samples pLM[ x, y ] that are constructed luma samples for Intra\_FromLuma prediction mode, with x = −1, y = −1..nT\*4−1 and x = 0..nT\*4−1, y=−1, are derived as following ordered steps:

1. For x = −1, y= 0..nT\*2−1 , if the chroma sample p[ x, y ] is marked as "not available for intra prediction" pLM[ x, y ] is marked as "not available for intra prediction", otherwise, pLM[ x, y ] is marked as "available for intra prediction" and the luma samples at the locations ( xB + x − 1, yB + y ) is assigned to pLM[ x, y ].
2. For x = 0..nT\*2−1, y=−1 , if the chroma sample p[ x, y ] is marked as "not available for intra prediction" pLM[ x, y ] is marked as "not available for intra prediction", otherwise, pLM[ x, y ] is marked as "available for intra prediction" and the luma samples at the locations ( xB + x , yB + y ) is assigned to pLM[ x, y ].
3. For x = −1, y = −1, if the chroma sample p[ x, y ] is marked as "not available for intra prediction" pLM[ x, y ] is marked as "not available for intra prediction", otherwise, pLM[ x, y ] is marked as "available for intra prediction" and the luma sample at the location ( xB + x , yB + y ) is assigned to pLM[ x, y ].

When at least one sample p[ x ][ y ] with x = −1, y = −1..nT \* 2 − 1 and x = 0..nT \* 2 − 1, y = −1 is marked as "not available for intra prediction", the reference sample substitution process for intra sample prediction in subclause is invoked with the samples p[ x ][ y ] with x = −1, y = −1..nT \* 2 − 1 and x = 0..nT \* 2 − 1, y = −1, nT, and cIdx as inputs, and the modified samples p[ x ][ y ] with x = −1, y = −1..nT \* 2 − 1 and x = 0..nT \* 2 − 1, y = −1 as output.

When chroma\_pred\_from\_luma\_enabled\_flag is equal to 1, cIdx is equal to 1 or 2, intraPredMode is equal to Intra\_FromLuma, and at least one luma sample pLM[ x, y ] with x = −1, y = −1..nT\*4−1 and x = 0..nT\*4−1, y = −1 is marked as "not available for intra prediction", the reference sample substitution process for intra sample prediction in subclause 7 is invoked with the samples pLM[ x, y ] with x = −1, y = −1..nT\*4−1 and x = 0..nT\*4−1, y = −1 and nT\*2 as input and the modified samples pLM[ x, y ] with x = −1, y = −1..nT\*4−1 and x = 0..nT\*4−1, y = −1 as output.

Depending on the value of predModeIntra, the following ordered steps apply:

1. When cIdx is equal to 0 or ChromaArrayType is equal to 3, the filtering process of neighbouring samples specified in is invoked with the sample array p and the transform block size nT as inputs, and the output is reassigned to the sample array p.
2. The intra sample prediction process according to predModeIntra applies as follows:
   * If predModeIntra is equal to INTRA\_PLANAR, the corresponding intra prediction mode specified in subclause 8.4.4.2.4 is invoked with the sample array p and the transform block size nT as inputs, and the output is the predicted sample array predSamples.
   * Otherwise, if predModeIntra is equal to INTRA\_DC, the corresponding intra prediction mode specified in subclause 8.4.4.2.5 is invoked with the sample array p, the transform block size nT, and the colour component index cIdx as inputs, and the output is the predicted sample array predSamples.
   * Otherwise, (predModeIntra is in the range of INTRA\_ANGULAR2..INTRA\_ANGULAR34), the corresponding intra prediction mode specified in subclause 8.4.4.2.6 is invoked with the intra prediction mode predModeIntra, the sample array p, the transform block size nT, and the colour component index cIdx as inputs, and the output is the predicted sample array predSamples.
   * Otherwise (intraPredMode is equal to Intra\_FromLuma), the intra prediction mode specified in subclause 7 is invoked with the sample location ( xB, yB ), the sample array pLM, and the transform block size nT as the inputs and the output are the predicted sample array predSamples.

##### Filtering process of neighbouring samples

Inputs to this process are:

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

– a variable nT specifying the transform block size.

Outputs of this process are the filtered samples pF[ x ][ y ], with x = −1, y = −1..nT \* 2 − 1 and x = 0..nT \* 2 − 1, y = −1.

The variable filterFlag is derived as follows:

– If one or more of the following conditions are true, filterFlag is set equal to 0:

* predModeIntra is equal to INTRA\_DC.
* nT is equal 4.
* intraPredMode is equal to Intra\_FromLuma

– Otherwise, the following applies:

* The variable minDistVerHor is set equal to Min( Abs( predModeIntra − 26 ), Abs( predModeIntra − 10 ) ).
* The variable intraHorVerDistThres[ nT ] is specified in .
* The variable filterFlag is derived as follows:
* If minDistVerHor is larger than intraHorVerDistThres[ nT ], filterFlag is set equal to 1.
* Otherwise, filterFlag is set equal to 0.

##### Specification of Intra\_FromLuma (35) prediction mode

Inputs to this process are:

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

– luma neighbouring samples pLM[ x, y ], with x = −1, y = 0... 2\*nT − 1; and x = 0…2\*nT − 1, y = − 1,

– a variable nT specifying the transform block size.

Output of this process are:

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

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

1. Variable k0 and the sample array pY’ are derived as:

k0 = Max( 0, BitDepthC + Log2( nT) − 14 ) (8‑58)

* 1. When ChromaArrayType is equal to 1,

– pY’[ x, −1 ] = ( PLM[ 2\*x−1, −1 ] + 2\*PLM[ 2\*x, −1 ] + PLM[ 2\*x+1, −1 ] + 2 ) >> 2, with x = 0..nT−1 (8‑59)

– pY’[ −1, y ] = ( PLM[ −1, 2\*y ] + PLM[ −1, 2\*y+1 ] ) >> 1, with y = 0.. nT −1 (8‑60)

– pY’[ x, y ] = ( recSamplesL[ 2\*x, 2\*y ] + recSamplesL[  2\*x,  2\*y+1 ] ) >> 1, with x, y = 0.. nT−1 (8‑61)

* 1. When ChromaArrayType is equal to 2,

– pY’[ x, −1 ] = ( PLM[ 2\*x−1, −1 ] + 2\*PLM[2\*x, −1 ] + PLM[ 2\*x+1, −1 ] + 2 ) >> 2, with x = 0.. nT−1 (8‑59)

– pY’[ −1, y ] = ( PLM[ −1, 2\*y ] + PLM[ −1, 2\*y+1 ] ) >> 1, with y = 0.. nT−1 (8‑60)

– pY’[ x, y ] = recSamplesL[ 2\*x, y ] , with x, y = 0..nT−1 (8‑61)

* 1. When ChromaArrayType is equal to 3,

– pY’[ x, −1 ] = PLM[ x, −1 ] , with x = 0..nT−1 (8‑59)

– pY’[ −1, y ] = PLM[ −1, y ], with y = 0..nT−1 (8‑60)

– pY’[ x, y ] = recSamplesL[x, y ], with x, y = 0..nT−1 (8‑61)

1. Variables l, c, ll, lc and k1 are derived as follows:

l =  (8‑62)

c =  (8‑63)

ll =  (8‑64)

lc =  (8‑65)

k1 = Log2( (2\*nS) >> k0 ) (8‑66)

1. Variables a, b and k are specified by the following pseudo-code:

avgY’ =  l >> k1  
errY’ =  l & ( ( 1 << k1 ) − 1 )  
avgC = c >> k1  
errC = c & ( ( 1 << k1 )  − 1 )  
a1 = lc − ( 2 \* nS \* avgY’ \* avgC + avgY’ \* errC + avgC \* errY’ )  
a2 = ll − ( 2 \* nS \* avgY’2 + 2 \* avgY’ \* errY’ )  
k2 = ( a1 = = 0) ? 0 : Max( 0, Floor( Log2( Abs( a1 ) ) ) − BitDepthC + 2 )  
k3 = ( a2 = = 0) ? 0 : Max( 0, Floor( Log2( Abs( a2 ) ) ) − 5 )  
k4 = k3 − k2 + BitDepthC − 2  
a1s = a1 >> k2  
a2s = a2 >> k3  
lmDiv = ( 1 << ( BitDepthC + 4 ) + a2s / 2 ) / a2s  
a2t = ( a2s < 32) ? 0 : lmDiv  
a2t = Clip1C ( a2t ) (8‑67)  
if( a2s < 32 )  
 a3 = 0  
else if( a2s >= 32 && k4 >= 0 )  
 a3 = ( a1s \* a2t ) >> k4 )  
else  
 a3 = ( a1s \* a2t ) << ( −k4 )  
a4 = Clip3( − 28, 28 − 1, a3 )  
a5 = a4 << 7  
k5 = ( a5 = = 0 ) ? 0 : k5 = Floor( Log2( Abs( a5 ) + ( Sign2( a5 ) − 1)/2 ) ) − 5  
k = 13 − k5  
a = a5 >> k5  
b = avgC − ( ( a \* avgY’ ) >> k )

1. The values of the prediction samples predSamples[ x, y ] are derived as:

predSamples[ x, y ] = Clip1C( ( ( pY’[ x, y ] \* a ) >> k ) + b ), with x, y = 0..nT−1 (8‑68)



#### Binarization process for intra\_chroma\_pred\_mode

Input to this process is a request for a binarization for the syntax element intra\_chroma\_pred\_mode.

Output of this process is the binarization of the syntax element.

The binarization for the syntax element intra\_chroma\_pred\_mode is specified in Table 9‑35.

Table 9‑35 – Specifcation of intra\_chroma\_pred\_mode depending on chroma\_pred\_from\_luma\_enabled\_flag

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Value of intra\_chroma\_pred\_mode** | **chroma\_pred\_from\_luma\_enabled\_flag = 1** | | **chroma\_pred\_from\_luma\_enabled\_flag = 0** | |
| **prefix** | **suffix** | **prefix** | **suffix** |
| 5 | 0 | n/a | n/a | n/a |
| 4 | 10 | n/a | 0 | n/a |
| 0 | 11 | 00 | 1 | 00 |
| 1 | 11 | 01 | 1 | 01 |
| 2 | 11 | 10 | 1 | 10 |
| 3 | 11 | 11 | 1 | 11 |



#### Derivation process for ctxTable, ctxIdx and bypassFlag

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| intra\_chroma\_pred\_mode[ ][ ]  (chroma\_pred\_from\_luma\_enabled\_flag == 1) | 0 | 1 | bypass | bypass | na | na |
| intra\_chroma\_pred\_mode[ ][ ]  (chroma\_pred\_from\_luma\_enabled\_flag == 0) | 0 | bypass | bypass | na | na | na |