# Draft Text Specification

The proposed text changes are based on the document JCTVC-L1005-v4-JCTVC-L1003\_v34.doc. We have started with the draft text provided in JCTVC-M0079 and modified it to account for the extensions proposed in JCTVC-M0288. The changes are marked in yellow. The changes with respect to M0079 are marked in cyan.

## Extension of residual DPCM to near-horizontal and near-vertical intra prediction modes

**8.4.4.2.6 Specification of intra prediction mode in the range of INTRA\_ANGULAR2.. INTRA\_ANGULAR34**

Inputs to this process are:

– the intra prediction mode predModeIntra,

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

– a variable nTbS specifying the transform block size,

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

Outputs of this process are the predicted samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1.

Figure 8‑2 illustrates the total 33 intra angles and Table 8‑4 specifies the mapping table between predModeIntra and the angle parameter intraPredAngle.



**Figure 8‑2 – Intra prediction angle definition (informative)**

**Table 8‑4 – Specification of intraPredAngle**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **predModeIntra** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** |
| **intraPredAngle** | - | 32 | 26 | 21 | 17 | 13 | 9 | 5 | 2 | 0 | −2 | −5 | −9 | −13 | −17 | −21 | −26 |
| **predModeIntra** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** | **32** | **33** | **34** |
| **intraPredAngle** | −32 | −26 | −21 | −17 | −13 | −9 | −5 | −2 | 0 | 2 | 5 | 9 | 13 | 17 | 21 | 26 | 32 |

Table 8‑5 further specifies the mapping table between predModeIntra and the inverse angle parameter invAngle.

**Table 8‑5 – Specification of invAngle**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **predModeIntra** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** |
| **invAngle** | −4096 | −1638 | −910 | −630 | −482 | −390 | −315 | −256 |
| **predModeIntra** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** |
| **invAngle** | −315 | −390 | −482 | −630 | −910 | −1638 | −4096 | - |

The variables intraPredAngleAdj and invAngleAdj are derived as follows:

– When ChromaArrayType is equal to 2 and cIdx is not equal to 0, the following applies:

* If intraPredMode is greater than 17, intraPredAngleAdj is set equal to intraPredAngle  >>  1) and invAngleAdj is set equal to invAngle  <<  1,
* Otherwise (intraPredMode is less than 18), the variables intraPredAngleAdj and invAngleAdj are derived as follows:
  + If Abs( intraPredAngle ) is greater than 16, intraPredAngleAdj is set equal to Sign( intraPredAngle ) \* 32 and invAngleAdj is set equal to −256,
  + Otherwise (Abs( intraPredAngle ) is less than 17), intraPredAngleAdj is set equal to intraPredAngle  <<  1 and when nTbS \* intraPredAngle  >>  5 is less than −1, invAngleAdj is set equal to invAngle  >>  1.

– Otherwise (ChromaArrayType is not equal to 2 or cIdx is equal to 0), intraPredAngleAdj is set equal to intraPredAngle and when nT \* intraPredAngle  >>  5 is less than −1, invAngleAdj is set equal to invAngle.

The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:

– If predModeIntra is equal or greater than 18, the following ordered steps apply:

1. The reference sample array ref[ x ] is specified as follows:

* The following applies:

ref[ x ] = p[ −1 + x ][ −1 ], with x = 0..nTbS (8‑47)

* If intraPredAngle is less than 0, the main reference sample array is extended as follows:
* When ( nTbS \* intraPredAngle )  >>  5 is less than −1,

ref[ x ] = p[ −1 ][ −1 + ( ( x \* invAngleAdj + 128 )  >>  8 ) ],  
 with x = −1..( nTbS \* intraPredAngleAdj )  >>  5 (8‑48)

* Otherwise,

ref[ x ] = p[ −1 + x ][ −1 ], with x = nTbS + 1..2 \* nTbS (8‑49)

1. The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:
   1. The index variable iIdx and the multiplication factor iFact are derived as follows:

iIdx = ( ( y + 1 ) \* intraPredAngleAdj )  >>  5 (8‑50)

iFact = ( ( y + 1 ) \* intraPredAngleAdj ) & 31 (8‑51)

* 1. Depending on the value of iFact, the following applies:
* If iFact is not equal to 0, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] =   
 ( ( 32 − iFact ) \* ref[ x + iIdx + 1 ] + iFact \* ref[ x + iIdx + 2 ] + 16 )  >>  5 (8‑52)

* Otherwise, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] = ref[ x + iIdx + 1 ] (8‑53)

* 1. When cu\_transquant\_bypass\_flag is not equal to 1, predModeIntra is equal to 26 (vertical), cIdx is equal to 0 and nTbS is less than 32, the following filtering applies with x = 0, y = 0..nTbS − 1:

predSamples[ x ][ y ] = Clip1Y( p[ x ][ −1 ] + ( ( p[ −1 ][ y ] − p[ −1 ][ −1 ] )  >>  1 ) ) (8‑54)

– Otherwise (predModeIntra is less than 18), the following ordered steps apply:

1. The reference sample array ref[ x ] is specified as follows:

* The following applies:

ref[ x ] = p[ −1 ][ −1 + x ], with x = 0..nTbS (8‑55)

* If intraPredAngle is less than 0, the main reference sample array is extended as follows:
* When ( nTbS \* intraPredAngle )  >>  5 is less than −1,

ref[ x ] = p[ −1 + ( ( x \* invAngle + 128 )  >>  8 ) ][ −1 ],  
 with x = −1..( nTbS \* intraPredAngle )  >>  5 (8‑56)

* Otherwise,

ref[ x ] = p[ −1 ][ −1 + x ], with x = nTbS + 1..2 \* nTbS (8‑57)

1. The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:
2. The index variable iIdx and the multiplication factor iFact are derived as follows:

iIdx = ( ( x + 1 ) \* intraPredAngle )  >>  5 (8‑58)

iFact = ( ( x + 1 ) \* intraPredAngle ) & 31 (8‑59)

1. Depending on the value of iFact, the following applies:

* If iFact is not equal to 0, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] =   
 ( ( 32 − iFact ) \* ref[ y + iIdx + 1 ] + iFact \* ref[ y + iIdx + 2 ] + 16 )  >>  5 (8‑60)

* Otherwise, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] = ref[ y + iIdx + 1 ] (8‑61)

1. When cu\_transquant\_bypass\_flag is not equal to 1, predModeIntra is equal to 10 (horizontal), cIdx is equal to 0 and nTbS is less than 32, the following filtering applies with x = 0..nTbS − 1, y = 0:

predSamples[ x ][ y ] = Clip1Y( p[ −1 ][ y ] + ( ( p[ x ][ −1 ] − p[ −1 ][ −1 ] )  >>  1 ) ) (8‑62)

## 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 following ordered steps apply:

1. The (nT)x(nT) array r is set equal to the (nT)x(nT) array of transform coefficients TransCoeffLevel[ xT ][ yT ][ cIdx ].
2. If CuPredMode[ xT ][ yT ] is equal to MODE\_INTRA and predModeIntra (predModeIntra is set equal to IntraPredModeY[ xT ][ yT ] if cIdx is equal to 0, otherwise, predModeIntra is set equal to IntraPredModeC) is between 22 and 30 or between 6 and 14 the intra residual transform-bypass decoding process as specified in subclause 8.6.6 is invoked with the size of the transform block nT, the intra prediction mode predModeIntra, and the (nT)x(nT) array r as the inputs, and the output is a modified version of the (nT)x(nT) array 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)

**8.6.6 Intra residual transform-bypass decoding process**

This process is invoked when cu\_transquant\_bypass\_flag is equal to 1, pred\_mode\_flag is equal to 1, and the applicable intra prediction mode is between 22 and 30 or between 6 and 14. 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 nT specifying the transform block size
* a variable predModeIntra specifying the intra prediction mode
* an (nT)x(nT) array r with elements r[ 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 a modified version of the (nT)x(nT) array r with elements r[ x ][ y ] containing the result of the intra residual transform-bypass decoding process.

Let f be a temporary (nT)x(nT) array with element f[ x ][ y ], which are derived by;

f[ x ][ y ] = r[ x ][ y ] with x=0..nT-1 and y=0..nT-1

Depending on predModeIntra, the following applies:

* If predModeIntra is between 22 and 30, the modified array r is derived by:

 with x=0..nT-1 and y=0..nT-1

Otherwise (predModeIntra is between 6 and 14), the modified array r is derived by:

 with x=0..nT-1 and y=0..nT-1

## Extension of residual DPCM to near-horizontal, near-vertical and diagonal intra prediction modes

**8.4.4.2.6 Specification of intra prediction mode in the range of INTRA\_ANGULAR2.. INTRA\_ANGULAR34**

Inputs to this process are:

– the intra prediction mode predModeIntra,

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

– a variable nTbS specifying the transform block size,

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

Outputs of this process are the predicted samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1.

Figure 8‑2 illustrates the total 33 intra angles and Table 8‑4 specifies the mapping table between predModeIntra and the angle parameter intraPredAngle.



**Figure 8‑2 – Intra prediction angle definition (informative)**

**Table 8‑4 – Specification of intraPredAngle**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **predModeIntra** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** |
| **intraPredAngle** | - | 32 | 26 | 21 | 17 | 13 | 9 | 5 | 2 | 0 | −2 | −5 | −9 | −13 | −17 | −21 | −26 |
| **predModeIntra** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** | **32** | **33** | **34** |
| **intraPredAngle** | −32 | −26 | −21 | −17 | −13 | −9 | −5 | −2 | 0 | 2 | 5 | 9 | 13 | 17 | 21 | 26 | 32 |

Table 8‑5 further specifies the mapping table between predModeIntra and the inverse angle parameter invAngle.

**Table 8‑5 – Specification of invAngle**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **predModeIntra** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** |
| **invAngle** | −4096 | −1638 | −910 | −630 | −482 | −390 | −315 | −256 |
| **predModeIntra** | **19** | **20** | **21** | **22** | **23** | **24** | **25** | **26** |
| **invAngle** | −315 | −390 | −482 | −630 | −910 | −1638 | −4096 | - |

The variables intraPredAngleAdj and invAngleAdj are derived as follows:

– When ChromaArrayType is equal to 2 and cIdx is not equal to 0, the following applies:

* If intraPredMode is greater than 17, intraPredAngleAdj is set equal to intraPredAngle  >>  1) and invAngleAdj is set equal to invAngle  <<  1,
* Otherwise (intraPredMode is less than 18), the variables intraPredAngleAdj and invAngleAdj are derived as follows:
  + If Abs( intraPredAngle ) is greater than 16, intraPredAngleAdj is set equal to Sign( intraPredAngle ) \* 32 and invAngleAdj is set equal to −256,
  + Otherwise (Abs( intraPredAngle ) is less than 17), intraPredAngleAdj is set equal to intraPredAngle  <<  1 and when nTbS \* intraPredAngle  >>  5 is less than −1, invAngleAdj is set equal to invAngle  >>  1.

– Otherwise (ChromaArrayType is not equal to 2 or cIdx is equal to 0), intraPredAngleAdj is set equal to intraPredAngle and when nT \* intraPredAngle  >>  5 is less than −1, invAngleAdj is set equal to invAngle.

The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:

– If predModeIntra is equal or greater than 18, the following ordered steps apply:

1. The reference sample array ref[ x ] is specified as follows:

* The following applies:

ref[ x ] = p[ −1 + x ][ −1 ], with x = 0..nTbS (8‑47)

* If intraPredAngle is less than 0, the main reference sample array is extended as follows:
* When ( nTbS \* intraPredAngle )  >>  5 is less than −1,

ref[ x ] = p[ −1 ][ −1 + ( ( x \* invAngleAdj + 128 )  >>  8 ) ],  
 with x = −1..( nTbS \* intraPredAngleAdj )  >>  5 (8‑48)

* Otherwise,

ref[ x ] = p[ −1 + x ][ −1 ], with x = nTbS + 1..2 \* nTbS (8‑49)

1. The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:
   1. The index variable iIdx and the multiplication factor iFact are derived as follows:

iIdx = ( ( y + 1 ) \* intraPredAngleAdj )  >>  5 (8‑50)

iFact = ( ( y + 1 ) \* intraPredAngleAdj ) & 31 (8‑51)

* 1. Depending on the value of iFact, the following applies:
* If iFact is not equal to 0, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] =   
 ( ( 32 − iFact ) \* ref[ x + iIdx + 1 ] + iFact \* ref[ x + iIdx + 2 ] + 16 )  >>  5 (8‑52)

* Otherwise, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] = ref[ x + iIdx + 1 ] (8‑53)

* 1. When cu\_transquant\_bypass\_flag is not equal to 1, predModeIntra is equal to 26 (vertical), cIdx is equal to 0 and nTbS is less than 32, the following filtering applies with x = 0, y = 0..nTbS − 1:

predSamples[ x ][ y ] = Clip1Y( p[ x ][ −1 ] + ( ( p[ −1 ][ y ] − p[ −1 ][ −1 ] )  >>  1 ) ) (8‑54)

– Otherwise (predModeIntra is less than 18), the following ordered steps apply:

1. The reference sample array ref[ x ] is specified as follows:

* The following applies:

ref[ x ] = p[ −1 ][ −1 + x ], with x = 0..nTbS (8‑55)

* If intraPredAngle is less than 0, the main reference sample array is extended as follows:
* When ( nTbS \* intraPredAngle )  >>  5 is less than −1,

ref[ x ] = p[ −1 + ( ( x \* invAngle + 128 )  >>  8 ) ][ −1 ],  
 with x = −1..( nTbS \* intraPredAngle )  >>  5 (8‑56)

* Otherwise,

ref[ x ] = p[ −1 ][ −1 + x ], with x = nTbS + 1..2 \* nTbS (8‑57)

1. The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:
2. The index variable iIdx and the multiplication factor iFact are derived as follows:

iIdx = ( ( x + 1 ) \* intraPredAngle )  >>  5 (8‑58)

iFact = ( ( x + 1 ) \* intraPredAngle ) & 31 (8‑59)

1. Depending on the value of iFact, the following applies:

* If iFact is not equal to 0, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] =   
 ( ( 32 − iFact ) \* ref[ y + iIdx + 1 ] + iFact \* ref[ y + iIdx + 2 ] + 16 )  >>  5 (8‑60)

* Otherwise, the value of the prediction samples predSamples[ x ][ y ] is derived as follows:

predSamples[ x ][ y ] = ref[ y + iIdx + 1 ] (8‑61)

1. When cu\_transquant\_bypass\_flag is not equal to 1, predModeIntra is equal to 10 (horizontal), cIdx is equal to 0 and nTbS is less than 32, the following filtering applies with x = 0..nTbS − 1, y = 0:

predSamples[ x ][ y ] = Clip1Y( p[ −1 ][ y ] + ( ( p[ x ][ −1 ] − p[ −1 ][ −1 ] )  >>  1 ) ) (8‑62)

## 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 following ordered steps apply:

1. The (nT)x(nT) array r is set equal to the (nT)x(nT) array of transform coefficients TransCoeffLevel[ xT ][ yT ][ cIdx ].
2. If CuPredMode[ xT ][ yT ] is equal to MODE\_INTRA and predModeIntra (predModeIntra is set equal to IntraPredModeY[ xT ][ yT ] if cIdx is equal to 0, otherwise, predModeIntra is set equal to IntraPredModeC) is 2, 18, 34, between 22 and 30 or between 6 and 14 the intra residual transform-bypass decoding process as specified in subclause 8.6.6 is invoked with the size of the transform block nT, the intra prediction mode predModeIntra, and the (nT)x(nT) array r as the inputs, and the output is a modified version of the (nT)x(nT) array 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)

**8.6.6 Intra residual transform-bypass decoding process**

This process is invoked when cu\_transquant\_bypass\_flag is equal to 1, pred\_mode\_flag is equal to 1, and the applicable intra prediction mode is 2, 18, 34, between 22 and 30 or between 6 and 14. 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 nT specifying the transform block size
* a variable predModeIntra specifying the intra prediction mode
* an (nT)x(nT) array r with elements r[ 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 a modified version of the (nT)x(nT) array r with elements r[ x ][ y ] containing the result of the intra residual transform-bypass decoding process.

Let f be a temporary (nT)x(nT) array with element f[ x ][ y ], which are derived by;

f[ x ][ y ] = r[ x ][ y ] with x=0..nT-1 and y=0..nT-1

Depending on predModeIntra, the following applies:

* If predModeIntra is between 22 and 30, the modified array r is derived by:

 with x=0..nT-1 and y=0..nT-1

Otherwise, if predModeIntra is between 6 and 14, the modified array r is derived by:

 with x=0..nT-1 and y=0..nT-1

Otherwise, if predModeIntra is 18, the modified array r is derived by:

 with x=0..nT-1 and y=0..nT-1

Otherwise, if predModeIntra is 34, the modified array r is derived by:

 with x=0..nT-1 and y=0..nT-1,

Otherwise (predModeIntra is 2), the modified array r is derived by:

 with x=0..nT-1 and y=0..nT-1.