**Proposed Specification for Working Draft**

* + - 1. **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 intraPredMode specifying the luma intra prediction mode,

– a variable nS specifying the prediction size.

– a variable cIdx specifying the chroma component of the current block,

Output of this process is:

– the predicted samples predSamples[ x, y ], with x, y =0..nS-1.

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

– The luma location (xBN, yBN ) is specified by

xBN = xB + x  (8‑14)

yBN = yB +y  (8‑15)

– Each sample p[ x, y ] with x = -1, y= -1..nS\*2-1  and x = 0, y = -1 is derived as follows

* If any of the following condition is true, the sample p[ x, y ] is marked as “not available for intra prediction”
  + the coding unit covering ( xBN, yBN ) is not available
  + the coding unit covering ( xBN, yBN ) is not coded as intra mode 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 ( xBN, yBN ) inside the treeblock tbAddrN is assigned to p[ x, y ].

When at least one sample p[ x, y ] with x = 1, y = 1..nS\*2 1 and x = 0..nS\*2 1, y = 1 is marked as “not available for intra prediction,” the reference sample substitution process for intra sample prediction in subclause 8.3.3.1.1 is invoked with the samples p[ x, y ] with x = 1, y = 1..nS\*2 1 and x = 0..nS\*2 1, y = 1 as input and the modified samples p[ x, y ] with x = 1, y = 1..nS\*2 1 and x = 0..nS\*2 1, y = 1 as output.

Depending on intraPredMode, the following ordered steps apply:

1. When cIdx is equal to 0, filtering process of neighbouring samples specified in 8.3.3.1.2 is invoked with the sample array p and the prediction size nS as the inputs and the output is reassigned to the sample array p.
2. Intra sample prediction process according to intraPredMode applies as follows:
   * One of the intra prediction modes specified in subclause 8.3.3.1.3 to 8.3.3.1.8 is invoked with the sample location ( xB, yB ), the sample array p, the prediction size nS and the chroma component index cIdx as the inputs and the output is the predicted sample array predSamples according to intraPredMode.

[Ed. (WJ): some functions do not use some input parameters. Above sentence should be improved]

* + - * 1. **Reference sample substitution process for intra sample prediction**

Inputs to this process are the reference samples p[ x, y ] with x = ‑1, y = ‑1..nS\*2‑1 and x = 0..nS\*2‑1, y = ‑1 for intra sample prediction.

Outputs of this process are the modified reference samples p[ x, y ] with x = ‑1, y = ‑1..nS\*2‑1 and x = 0..nS\*2‑1, y = ‑1 for intra sample prediction.

The values of the samples p[ x, y ] with x = ‑1, y = ‑1..nS\*2‑1 and x = 0..nS\*2‑1, y = ‑1 are modified as follows:

* If all samples p[ x, y ] with x = ‑1, y = ‑1..nS\*2‑1 and x = 0..nS\*2‑1, y = ‑1 are marked as “not available for intra prediction,” the value ( 1 << ( BitDepthY ‑ 1 ) ) is substituted for the values of all samples p[ x, y ].
* Otherwise (at least one but not all samples p[ x, y ] are marked as “not available for intra prediction”), the following steps are performed for each sample p[ x0, y0 ] with x0 = ‑1, y0 = ‑1..nS\*2‑1 and x0 = 0..nS\*2‑1, y0 = ‑1 marked as "not available for intra prediction":
* Let the variables q and r be initially set to -1.
* The variable q is modified as follows:
  + If x0 is equal to ‑1, a reference sample p[ x, y ] marked as “available for intra prediction” is identified by searching sequentially among p[ x, y ] starting from x = ‑1, y = y0+1 to x = ‑1, y = nS\*2‑1. As soon as a sample marked as “available for intra prediction” is found, the search is terminated and the value of p[ x, y ] is assigned to q. [Ed.: (WJ) is there any elegant way rather than searching?]
  + Otherwise (x0 is not equal to ‑1), a sample marked as “available for intra prediction” is identified among the samples p[ x, y ] by searching sequentially starting from x = x0‑1, y = -1 to x = ‑1, y = ‑1. When a sample marked as “available for intra prediction” is not found among p[ x, y ] with x = x0‑1..‑1, y = -1, the search is continued among p[ x, y ] sequentially starting from x = ‑1, y = 0 to x = ‑1, y = nS\*2‑1. As soon as a sample marked as “available for intra prediction” is found, the search is terminated and the value of p[ x, y ] is assigned to q. [Ed.: (WJ) is there any elegant way rather than searching?]
* The variable r is modified as follows:
  + If x0 is equal to ‑1, a sample marked as “available for intra prediction” is identified among the samples p[ x, y ] by searching sequentially starting from x = ‑1, y = y0‑1 to x = ‑1, y = ‑1. When a sample marked as “available for intra prediction” is not found among p[ x, y ] with x = ‑1, y = y0‑1..-1, the search is continued among p[ x, y ] sequentially starting from x = 0, y = ‑1 to x = nS\*2‑1, y = ‑1. As soon as a sample marked as “available for intra prediction” is found, the search is terminated and the value of p[ x, y ] is assigned to r. [Ed.: (WJ) is there any elegant way rather than searching?]
  + Otherwise (x0 is not equal to ‑1), a sample marked as “available for intra prediction” is identified among the samples p[ x, y ] by searching sequentially starting from x = x0+1, y = -1 to x = nS\*2‑1, y = ‑1. As soon as a sample marked as “available for intra prediction” is found, the search is terminated and the value of p[ x, y ] is assigned to r. [Ed.: (WJ) is there any elegant way rather than searching?]
* The value of the sample p[ x0, y0 ] is modified as follows:
  + If q is not equal to ‑1 and r is not equal to ‑1, the value ( ( q + r + 1 ) >> 1 ) is substituted for the value of p[ x0, y0 ].
  + Otherwise, if q is not equal to ‑1, the value of q is substituted for the value of p[ x0, y0 ].
  + Otherwise (q is equal to ‑1 and r is not equal to ‑1), the value of r is substituted for the value of p[ x0, y0 ].

All samples p[ x, y ] with x = ‑1, y = ‑1..nS\*2‑1 and x = 0..nS\*2‑1, y = ‑1 are marked as “available for intra prediction.”

* + - * 1. **Filtering process of neighbouring samples**

Inputs to this process are:

– neighbouring samples p[ x, y ], with x, y = -1..2\*nS-1,

– a variable nS specifying the prediction size.

Output of this process is:

– filtered samples pF[ x, y ],. with x, y = -1..2\*nS-1.

**Table 8‑6 – Specification of intraFilterType[ nS ][ IntraPredMode ] for various prediction unit sizes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **IntraPredMode** | **intraFilterType**  **for nS = 4** | **intraFilterType**  **for nS = 8** | **intraFilterType**  **for nS = 16** | **intraFilterType**  **for nS = 32** | **intraFilterType**  **for nS = 64** |
| 0-2 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 | 0 |
| 4, 5 | 0 | 0 | 1 | 1 | 0 |
| 6 | 0 | 1 | 1 | 1 | 0 |
| 7, 8 | 0 | 0 | 1 | 1 | 0 |
| 9 | 0 | 1 | 1 | 1 | 0 |
| 10-20 | 0 | 0 | 1 | 1 | 0 |
| 21, 22 | 0 | 0 | 0 | 1 | 0 |
| 23-28 | 0 | 0 | 1 | 1 | 0 |
| 29, 30 | 0 | 0 | 0 | 1 | 0 |
| 31-33 | 0 | 0 | 1 | 1 | 0 |
| 34 | 0 | 1 | 1 | 1 | 0 |
| 35 | n/a | n/a | n/a | n/a | n/a |

Filtered sample array pF[ x, y ] with x = -1..nS\*2-1 and y = -1..nS\*2-1  are derived as follows:

– When intraFilterType[ nS ][ IntraPredMode ] is equal to 1, the following applies:

pF[ -1, nS\*2-1 ] = p[ -1, nS\*2-1 ] (8‑17)

pF[ nS\*2-1, -1 ] = p[ nS\*2-1, -1 ] (8‑18)

pF[ -1, y ] = ( p[ -1, y+1 ] + 2\*p[ -1, y ] + p[ -1, y-1 ] + 2 ) >> 2 for y = nS\*2-2..0 (8‑19)

pF[ -1, -1] = ( p[ -1, 0 ] + 2\*p[ -1, -1] + p[ 0, -1 ] + 2) >> 2 (8‑20)

pF[ x, -1 ] = ( p[ x-1, -1 ] + 2\*p[ x, -1 ] + p[ x+1, -1 ] + 2 ) >> 2 for x = 0..nS\*2-2 (8‑21)

* + - * 1. **Specification of Intra\_Vertical prediction mode**

Inputs to this process are:

– neighbouring samples p[ x, y ], with x, y = -1..2\*nS-1,

– a variable nS specifying the prediction size

Output of this process is:

– predicted samples predSamples[ x, y ], with x, y =0..nS-1.

This intra prediction mode is invoked when intraPredMode is equal to 0.

The values of the prediction samples predSamples[ x, y ], with x, y = 0..nS-1, are derived by

predSamples[ x, y ] = p[ x, -1 ], with x, y = 0..nS-1 (8‑30)

* If cIdx is equal to 0
* for(y=0; y<nS;y++)

gradientVal = p[-1,y]-p[-1,-1]

predSamples[0,y] = Clip(predSamples[0,y] + ((gradientVal + 1)>>1))

predSamples[1,y] = Clip(predSamples[1,y] + ((gradientVal + 2)>>2))

* + - * 1. **Specification of Intra\_Horizontal prediction mode**

Inputs to this process are:

– neighbouring samples p[ x, y ], with x, y = -1..2\*nS-1,

– a variable nS specifying the prediction size.Output of this process is:

– predicted samples predSamples[ x, y ], with x, y =0..nS-1.

This intra prediction mode is invoked when intraPredMode is equal to 1.

The values of the prediction samples predSamples[ x, y ], with x, y = 0..nS-1, are derived by

predSamples[ x, y ] = p[ -1, y ], with x, y = 0..nS-1 (8‑31)

* If cIdx is equal to 0
* for(x=0; x<nS;x++)

gradientVal = p[x,-1]-p[-1,-1]

predSamples[x,0] = Clip(predSamples[x,0] + ((gradientVal + 1)>>1))

predSamples[x,1] = Clip(predSamples[x,1] + ((gradientVal + 2)>>2))

* + - * 1. **Specification of Intra\_DC prediction mode**

Inputs to this process are:

– neighbouring samples p[ x, y ], with x, y = -1..2\*nS-1,

– a variable nS specifying the prediction block size.

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

Output of this process is:

– predicted samples predSamples[ x, y ], with x, y =0..nS-1.

This intra prediction mode is invoked when intraPredMode is equal to 2.

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

1. A variable DCVal is derived as:

DCVal = , with x, y = 0..nS-1 (8‑32)  
where k=log2(nS)

1. Depending on the prediction block size nS and the chroma component index cIdx, the following applies.

* If cIdx is equal to 0 and nS is less than 64, the following applies.

predSamples[ 0, 0 ] = ( 1\*p[ -1, 0 ] + 2\*DCVal + 1\*p[ 0, -1 ] + 2 ) >> 2 (8‑32)  
predSamples[ x, 0 ] = ( 1\*p[ x, -1 ] + 3\*DCVal + 2 ) >> 2, with x = 1..nS-1 (8‑32)  
predSamples[ 0, y ] = ( 1\*p[ -1, y ] + 3\*DCVal + 2 ) >> 2, with y = 1..nS-1 (8‑32)  
predSamples[ x, y ] = DCVal, with x, y = 1..nS-1 (8‑32)

* Otherwise, the prediction samples predSamples[ x, y ] are derived as

predSamples[ x, y ] = DCVal, with x, y = 0..nS-1 (8‑32)

* + - * 1. **Specification of Intra\_Angular prediction mode**

Inputs to this process are:

– neighbouring samples p[ x, y ], with x, y = -1..2\*nS-1,

– a variable nS specifying the prediction size.

Output of this process is:

– predicted samples predSamples[ x, y ], with x, y =0..nS-1.

This intra prediction mode is invoked when intraPredMode is in the range of 3..33.

Table 8‑6 specifies the mapping table between intraPredMode and the rearranged intra prediction order intraPredOrder.

**Table 8‑7 – Specification of intraPredOrder**

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

Figure 8‑2 illustrates the total 34 intra angles and Table 8‑7 specifies the mapping table between intraPredOrder and the angle parameter intraPredAngle.



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

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

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

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

**Table 8‑9 – Specification of invAngle**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **intraPredOrder** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| **invAngle** | -256 | -315 | -390 | -482 | -630 | -910 | -1638 | -4096 |
| **intraPredOrder** | **18** | **19** | **20** | **21** | **22** | **23** | **24** | **25** |
| **invAngle** | -315 | -390 | -482 | -630 | -910 | -1638 | -4096 | - |

The reference pixel array refMain[ x ], with x=-nS..2\*nS is specified as follows.

– If intraPredOrder is less than 18,

refMain[ x ] = p[ -1+x, -1 ], with x=0..nS (8‑36)

* If intraPredAngle is less than 0,

refMain[ x ] = p[ -1, -1+( ( x\*invAngle+128 )>>8 ) ], with x=( nS\*intraPredAngle ) >>5..-1 (8‑37)

* Otherwise,

refMain[ x ] = p[ -1+x, -1 ], with x=nS+1..2\*nS (8‑38)

Otherwise,

refMain[ x ] = p[ -1, -1+x ], with x=0..nS (8‑39)

* If intraPredAngle is less than 0,

refMain[ x ] = p[ -1+( ( x\*invAngle+128 )>>8 ), -1 ], with x=( nS\*intraPredAngle ) >>5..-1 (8‑40)

* Otherwise,

refMain[ x ] = p[ -1, -1+x ], with x=nS+1..2\*nS (8‑41)

The values of the prediction samples predSamples[ x, y ], with x, y = 0..nS-1 are derived by the following procedures.

– The index variable iIdx and the multiplication factor iFact are derived by

iIdx = ( ( y + 1 )\*intraPredAngle ) >> 5 (8‑42)

iFact = ( ( y + 1 )\*intraPredAngle ) && 31 (8‑43)

– 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 by

predSamples[ x, y ] = ( ( 32 – iFact )\*refMain[ x+iIdx+1 ] + iFact\*refMain[ x+iIdx+2] + 16 ) >> 5 (8‑44)

* Otherwise, the value of the prediction samples predSamples[ x, y ] is derived by

predSamples[ x, y ] = refMain[ x+iIdx+1 ] (8‑45)

* If cIdx is equal to 0
* If (( intraPredOrder >=1) && ( intraPredOrder <=5))
* For(y=0;y<nS;y++)

predSamples[0,y]=(6\*predSamples[0,y]+2\*p[-1,y-1]+4)>>3

* If (( intraPredOrder ==10) || ( intraPredOrder ==11))
* iIdx = ( ( y + 1 )\*intraPredAngle ) >> 5
* iFact = ( ( y + 1 )\*intraPredAngle ) && 31
* If iFact is not equal to 0, the value of the prediction samples pRefSamples[ -1, y ] is derived by

pRefSamples[ -1, y ] = ( ( 32 – iFact )\*refMain[ iIdx ] + iFact\*refMain[ iIdx+1] + 16 ) >> 5

* Otherwise, the value of the prediction samples pRefSamples[ -1, y ] is derived by

pRefSamples[ -1, y ] = refMain[ iIdx ]

* For(y=0;y<nS;y++)

predSamples[0,y] = Clip(predSamples[0,y] + ((p[-1,y]-pRefSamples[-1,y]+2)>>2))

* If (( intraPredOrder >=13) && ( intraPredOrder <=16))
* For(y=0;y<nS;y++)

predSamples[0,y]=(6\*predSamples[0,y]+2\*p[-1,y+1]+4)>>3

* If IntraPredOrder is equal to 17
* For(y=0;y<nS;y++)

predSamples[0,y]=(8\*predSamples[0,y]+8\*p[-1,y+1]+8)>>4

predSamples[1,y]=(12\*predSamples[1,y]+4\*p[-1,y+2]+8)>>4

predSamples[2,y]=(14\*predSamples[2,y]+2\*p[-1,y+3]+8)>>4

predSamples[3,y]=(15\*predSamples[3,y]+p[-1,y+4]+8)>>4

* If (( intraPredOrder >=18) && ( intraPredOrder <=21))
* For(x=0;x<nS;x++)

predSamples[x,0]=(6\*predSamples[x,0]+2\*p[x-1,-1]+4)>>3

* If (( intraPredOrder ==26) || ( intraPredOrder ==27))
* iIdx = ( ( x + 1 )\*intraPredAngle ) >> 5
* iFact = ( ( x + 1 )\*intraPredAngle ) && 31
* If iFact is not equal to 0, the value of the prediction samples pRefSamples[ x, -1 ] is derived by

pRefSamples[ x, -1 ] = ( ( 32 – iFact )\*refMain[ iIdx ] + iFact\*refMain[ iIdx+1] + 16 ) >> 5

* Otherwise, the value of the prediction samples pRefSamples[ x, -1 ] is derived by

pRefSamples[ x, -1 ] = refMain[ iIdx ]

* For(x=0;x<nS;x++)

predSamples[x,-1] = Clip(predSamples[x,-1] + ((p[x,-1]-pRefSamples[x,-1]+2)>>2))

* If (( intraPredOrder >=29) && ( intraPredOrder <=32))
* For(x=0;x<nS;x++)

predSamples[x,0]=(6\*predSamples[x,0]+2\*p[x+1,-1]+4)>>3

* If IntraPredOrder is equal to 33
* For(x=0;x<nS;x++)

predSamples[x,0]=(8\*predSamples[x,0]+8\*p[x+1,-1]+8)>>4

predSamples[x,1]=(12\*predSamples[x,1]+4\*p[x+2,-1]+8)>>4

predSamples[x,2]=(14\*predSamples[x,2]+2\*p[x+3,-1]+8)>>4

predSamples[x,3]=(15\*predSamples[x,3]+p[x+4,-1]+8)>>4

* + - * 1. **Specification of Intra\_Planar prediction mode**

Inputs to this process are:

– neighbouring samples p[ x, y ], with x, y = -1..2\*nS-1,

– a variable nS specifying the prediction size,

Output of this process is:

– predicted samples predSamples[ x, y ], with x, y =0..nS-1.

This intra prediction mode is invoked when intraPredMode is equal to 34.

The values of the prediction samples predSamples[ x, y ], with x, y = 0..nS-1, are derived by

predSamples[ x, y ] = (  
 ( nS – 1 – x ) \* p[ -1, y ] + ( x + 1 ) \* p[ nS, -1 ] +   
 ( nS – 1 – y ) \* p[ x ,-1 ] + ( y + 1 ) \* p[ -1, nS ] + nS ) >> ( k + 1 ) (8‑45)  
with x, y = 0..nS-1 where k = log2( nS )

* If cIdx is equal to 0 and nS is less than 64

predSamples[ 0, 0 ] = ( 1\*p[ -1, 0 ] + 2\*predSamples[0, 0] + 1\*p[ 0, -1 ] + 2 ) >> 2

* for(x=0; x<nS;x++)

predSamples[ x, 0 ] = ( 1\*p[ x, -1 ] + 3\*predSamples[x, 0] + 2 ) >> 2

* for(y=0; y<nS;y++)

predSamples[ 0, y ] = ( 1\*p[ -1, y ] + 3\*predSamples [0, y] + 2 ) >> 2