### Deblocking filter process

…

The deblocking filter process shall be applied to all prediction unit edges and transform unit edges of a picture, except edges at the boundary of the picture and any edges for which the deblocking filter process is disabled by disable\_deblocking\_filter\_idc.

For luma component, for the transform units and prediction units with edges smaller than 8 samples in either vertical or horizontal direction, only the edges lying on the 8x8 sample grid are filtered. If the luma component of one edge is filtered by luma filtering, the corresponding chroma component of this edge is also filtered by chroma filtering.

If disable\_deblocking\_filter\_idc is equal to 1, the deblocking filter process is simply omitted otherwise the following ordered steps apply for each coding unit with the same order as decoding process.

#### **…**

#### **Filtering process for coding unit**

Inputs of this process are:

– a luma location ( xC, yC ) specifying the top-left luma sample of the current coding unit relative to the top left luma sample of the current picture,

– a variable log2CUSize specifying the coding unit size,

– an array bS specifying the boundary filtering strength.

Output of this process is:

– modified reconstruction of the picture.

The filtering process for luma edges in the current coding unit consists of the following ordered steps:

1. The variable nD is set equal to 1 << ( log2CUSize – 3 ).
2. All elements of the three-dimensional array of size (2)x(nD)x(nD), dEdge are initialized to zero.
3. All elements of the three-dimensional array of size (2)x(nD)x(1<<log2CUSize), dSample are initialized to zero.
4. All elements of the three-dimensional array of size (2)x(nD)x(nD), bStrength ae initialized to zero.
5. For xDk set equal to xC+( k << 3 ), k=0..nD – 1, the following applies:

* For yDm set equal to yC+( m << 3 ), m=0..nD – 1, the following ordered steps apply:
  + 1. Boundary filtering strength bSVer is derived as follows:

bSVer = Max( bS[ 0 ][ xDk ][ yDm + i ] ) for i = 0..7 (8‑428)

* + 1. bStrength[1][k][m] is set equal to bSVer.
    2. Decision process for luma block edge in subclause 8.6.1.4.1 is invoked with the luma location of the coding unit ( xC, yC ), the luma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 1, and the boundary filtering strength bSVer as inputs and the decision dEdge[1][k][m], an array dS of size (8) and a variable d as outputs.
    3. dSample[1][k][(m<<3)+i] is set equal to dS[i] for i=0..7, and diff[1][k][m] is set equal to d.
    4. Boundary filtering strength bSHor is derived as follows:

bSHor = Max( bS[ 1 ][ xDk + i ][ yDm ] ) for i = 0..7 (8‑428)

* + 1. bStrength[0][k][m] is set equal to bSHor.
    2. Decision process for luma block edge in subclause 8.6.1.4.1 is invoked with the luma location of the coding unit ( xC, yC ), the luma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 0, the boundary filtering strength bSHor as inputs, the decision dEdge[0][k][m], an array dS of size (8) and a variable d as outputs.
    3. dSample[0][m][(k<<3)+i] is set equal to dS[i] for i=0..7, and diff[0][k][m] is set equal to d.

1. For xDk set equal to xC+( k << 3 ), k=0..nD - 1, the following applies:

* For yDm set equal to yB+( m << 3 ), m=0..nD – 1, the following ordered steps apply:
  + 1. dS[i] is set equal to dSample[1][k][(m<<3)+i] for i=0..7.
    2. Filtering process for luma block edge in subclause 8.6.1.4.2 is invoked with the luma location of the coding unit ( xC, yC ), the luma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 1, the boundary filtering strength bStrength[1][k][m], the decision dEdge[1][k][m], and the array of size (8), dS as inputs and the modified luma picture buffer as outputs.

1. For yDm set equal to yC+( m << 3 ), m=0..nD - 1, the following applies:

* For xDk set equal to xC+( k << 3 ), k=0..nD – 1, the following ordered steps apply:
  + 1. If xDk is equal to 0, the parameter xPOS is set equal to 1. If xDk is equal to xB+( ( nD – 1) << 3 ) xPOS is set equal to 2. Otherwise xPOS is set to 0.
    2. dS[i] is set equal to dSample[0][m][ (k << 3) + i ] for i = 0..7.
    3. Filtering process for luma block edge in subclause 8.6.1.4.2 is invoked with the luma location of the coding unit ( xC, yC ), the luma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 0, the boundary filtering strength bStrength[0][k][m], the decision dEdge[0][k][m], and the array of size (8), dS, xPOS, dSL[m][], dEL[m], bSL[m], and tCL[m], as inputs and the modified luma picture buffer as output.
* The elements of the two dimensional array of size (3)x(nD), dSL are set as follows. dSL[m][0], dSL[m][1], and dSL[m][2], are set equal to dS[5], dS[6] and dS[7].
* The elements of the array of size (nD), dEL are set as follows. dEL[m] is set equal to dEdge[0][k][m].
* The elements of the array of size (nD), bSL are set as follows. bSL[m] is set equal to bStrength[0][k][m].
* The elements of the array of size (nD), tC are set as follows. tCL[m] is set equal to tc.

The filtering process for chroma edges in the current coding unit consists of the following ordered steps:

1. The variable nD is set equal to 1 << ( Max( log2CUSize, 4 ) – 4 ).
2. For xDk set equal to ( xC / 2 )+( k << 2 ), k=0..nD – 1, the following applies:

* For yDm set equal to ( yC / 2)+( m << 2 ), m=0..nD – 1, the following ordered steps apply:

1. Boundary filtering strength bSVer is derived as follows:

bSVer = bS[ 0 ][ xDk\*2 ][ yDm\*2 ] (8‑430)

1. Filtering process for chroma block edge in subclause 8.6.1.4.3 is invoked with the chroma location of the coding unit ( xC/2, yC/2 ), the chroma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 1, a chroma component index cIdx set equal to 1, the boundary filtering strength bSVer and the luma sample difference diff[1][k][m] as inputs and the modified chroma picture buffer as output.
2. Filtering process for chroma block edge in subclause 8.6.1.4.3 is invoked with the chroma location of the coding unit ( xC/2, yC/2 ), the chroma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 1, a chroma component index cIdx set equal to 2 and the boundary filtering strength bSVer and the luma sample difference diff[1][k][m] as inputs and the modified chroma picture buffer as output.
3. For yDm set equal to ( yC / 2 )+( m << 2 ), m=0..nD – 1, the following applies:

* For xDk set equal to ( xC / 2 )+( k << 2 ), k=0..nD – 1, the following ordered steps apply:

1. If xDk is equal to 0, the parameter xPOS is set equal to 1. If xDk is equal to xB+( ( nD\*2 – 1) << 2 ) xPOS is set equal to 2. Otherwise xPOS is set to 0.
2. Boundary filtering strength bSHor is derived as follows:

bSHor = bS[ 1 ][ xDk\*2 ][ yDm\*2 ] (8‑431)

1. Filtering process for chroma block edge in subclause 8.6.1.4.3 is invoked with the chroma location of the coding unit ( xC/2, yC/2 ), the chroma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 0, a chroma component index cIdx set equal to 1 and the boundary filtering strength bSHor, xPOS, bSL[m] and tCL[m] and the luma sample difference diff[0][k][m] as inputs and the modified chroma picture buffer as output.
2. Filtering process for chroma block edge in subclause 8.6.1.4.3 is invoked with the chroma location of the coding unit ( xC/2, yC/2 ), the chroma location of the block ( xDk, yDm ), a variable verticalEdgeFlag set equal to 0, a chroma component index cIdx set equal to 2 and the boundary filtering strength bSHor, xPOS, bSL[m] and tCL[m] and the luma sample difference diff[1][k][m] as inputs and the modified chroma picture buffer as output.

##### Decision process for luma block edge

Inputs of this process are:

– a luma location ( xC, yC ) specifying the top-left luma sample of the current coding unit relative to the top left luma sample of the current picture,

– 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 coding unit,

– a variable verticalEdgeFlag,

– a variable bS specifying the boundary filtering strength,

Output of this process is:

– a variable dE containing a decision,

– one-dimensional array of size (8), dS containing decisions.

– a variable d

Let s’ represent the luma sample array recPictureL of the current picture.

A variables β is specified as Table 8‑13 with luma quantization parameter qPL as input.

A variable tC is specified as follows:

– If bS is greater than 2, the variable tC is specified specified as Table 8‑13 with luma quantization parameter ( qPL + 4 ) as input,

– Otherwise (bS is equal or less than 2), the variable tC is specified specified as Table 8‑13 with luma quantization parameter qPL as input.

Depending on verticalEdgeFlag, the following applies:

– If verticalEdgeFlag is equal to 1, the following ordered steps apply:

1. The sample values pi,k and qi,k with i = 0..3 and k = 2,5 are derived as follows:

qi,k = s’[ xC + xB +i, yC + yB + k ] (8‑447)

pi,k = s’[ xC + xB – i – 1, yC + yB + k ] (8‑448)

1. The variable d is derived as follows:

d = | p2,2 – 2\*p1,2 + p0,2 | + | q2,2 – 2\*q1,2 + q0,2 | + | p2,5 – 2\*p1,5 + p0,5 | + | q2,5 – 2\*q1,5 + q0,5 | (8‑449)

1. The variable dE is set equal to 0.
2. If bS is not equal to 0 and d is less than β, the following ordered steps apply:
3. for each sample location ( xC + xB, yC + yB + k ), k = 0..7, the following ordered steps apply:
   1. The decision process for a luma sample specified in subclause 8.6.1.4.4 is invoked with sample values pi,k, qi,k with i = 0..3, the boundary filtering strength bS and the variables d, β and tC as inputs and a decision dSam as output.
   2. The variable dS[k] is set equal to dSam
4. The variable dE is set equal to 1.

– Otherwise (verticalEdgeFlag is equal to 0), the following ordered steps apply:

1. The sample values pi,k and qi,k with i = 0..3 and k = 2,5 are derived as follows:

qi,k = s’[ xC + xB +k, yC + yB + i ] (8‑435)

pi,k = s’[ xC + xB +k, yC + yB – i – 1 ] (8‑436)

1. The variable d is derived as follows:

d = | p2,2 – 2\*p1,2 + p0,2 | + | q2,2 – 2\*q1,2 + q0,2 | + | p2,5 – 2\*p1,5 + p0,5 | + | q2,5 – 2\*q1,5 + q0,5 | (8‑437)

1. The variable dE is set equal to 0.
2. If bS is not equal to 0 and d is less than β, the following ordered steps apply:
3. For each sample location ( xC + xB + k, yC + yB ), k = 0..7, the following ordered steps apply:
   1. The decision process for a luma sample specified in subclause 8.6.1.4.4 is invoked with sample values pi,k, qi,k with i = 0..3, the boundary filtering strength bS and the variables d, β and tC as inputs and a decision dSam as output.
   2. The variable dS[k] is set equal to dSam.
4. The variable dE is set equal to 1.

##### Filtering process for luma block edge

Inputs of this process are:

– a luma location ( xC, yC ) specifying the top-left luma sample of the current coding unit relative to the top left luma sample of the current picture,

– 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 coding unit,

– a variable verticalEdgeFlag,

– a variable bS specifying the boundary filtering strength,

– a variable dE containing a decision,

– one-dimensional array of size (8), dS containing decisions,

– a variable bSL,

– a variable tCL,

Output of this process is:

– modified reconstruction of the picture.

Let s’ represent the luma sample array recPictureL of the current picture.

A variables β is specified as Table 8‑11 with luma quantization parameter qPL as input.

A variable tC is specified as follows:

– If bS is greater than 2, the variable tC is specified as Table 8‑11 with luma quantization parameter ( qPL + 4 ) as input,

– Otherwise (bS is equal or less than 2), the variable tC is specified as Table 8‑11 with luma quantization parameter qPL as input.

Depending on verticalEdgeFlag, the following applies:

– If verticalEdgeFlag is equal to 1, the following ordered steps apply:

1. The sample values pi,k and qi,k with i = 0..3 and k = 0..7 are derived as follows:

qi,k = s’[ xC + xB +i, yC + yB + k ] (8‑432)

pi,k = s’[ xC + xB – i – 1, yC + yB + k ] (8‑433)

1. If dE is not equal to 0, for each sample location ( xC + xB, yC + yB + k ), k = 0..7, the following ordered steps apply:
2. The filtering process for a luma sample specified in subclause 8.6.1.4.5 is invoked with sample values pi,k, qi,k with i = 0..3, the decision dS[k], the boundary filtering strength bS and the variable tC as inputs and the number of filtered samples nD and the filtered sample values pi’ and qi’ as outputs.
3. The filtered sample values pi’ and qi’ with i = 0..nD – 1 replace the corresponding samples inside the sample array s’ as follows:

s’[ xC + xB +i, yC + yB + k ] = qi’ (8‑435)

s’[ xC + xB – i – 1, yC + yB + k ] = pi’ (8‑436)

– Otherwise (verticalEdgeFlag is equal to 0), the following ordered steps apply:

1. If xPOS is equal to 1, the parameters ks and ke are set to -3 and 4 respectively. If xD is equal to 2, the parameters ks and ke are set to 0 and 4 respectively. Otherwise ks and ke are set to 0 and 7 respectively.
2. The sample values pi,k and qi,k with i = 0..3 and k = ks..ke are derived as follows:

qi,k = s’[ xC + xB +k, yC + yB + i ] (8‑437)

pi,k = s’[ xC + xB +k, yC + yB – i – 1 ] (8‑438)

1. If xPOS is less than 0 and dEL is not equal to 0, for each sample location ( xC + xB + k, yC + yB ), k = -3..-1, the following ordered steps apply:
2. The filtering process for a luma sample specified in subclause 8.6.1.4.5 is invoked with sample values pi,k, qi,k with i = 0..3, decision dSL[k+3], the boundary filtering strength bSL and the variable tCL as inputs and the number of filtered samples nD and the filtered sample values pi’ and qi’ as outputs.
3. The filtered sample values pi’ and qi’ with i = 0..nD – 1 replace the corresponding samples inside the sample array s’ as follows:

s’[ xC + xB +k, yC + yB + i ] = qi’ (8‑459)

s’[ xC + xB +k, yC + yB – i – 1 ] = pi’ (8‑460)

1. If dE is not equal to 0, for each sample location ( xC + xB + k, yC + yB ), k = 0.. ke , the following ordered steps apply:
2. The filtering process for a luma sample specified in subclause 8.6.1.4.5 is invoked with sample values pi,k, qi,k with i = 0..3, decision dS[k], the boundary filtering strength bS and the variable tC as inputs and the number of filtered samples nD and the filtered sample values pi’ and qi’ as outputs.
3. The filtered sample values pi’ and qi’ with i = 0..nD – 1 replace the corresponding samples inside the sample array s’ as follows:

s’[ xC + xB +k, yC + yB + i ] = qi’ (8‑461)

s’[ xC + xB +k, yC + yB – i – 1 ] = pi’ (8‑462)

##### Filtering process for chroma block edge

[Ed.: (WJ) cIdx cannot be 0 here]

Inputs of this process are:

– a luma location ( xC, yC ) specifying the top-left chroma sample of the current coding unit relative to the top left chroma sample of the current picture,

– a luma location ( xB, yB ) specifying the top-left chroma sample of the current block relative to the top left chroma sample of the current coding unit,

– a variable verticalEdgeFlag,

– a variable bS specifying the boundary filtering strength,

– a variable cIdx specifying the chroma component index.

– a variable xPOS,

– a variable bSL,

– a variable tCL

– a variable d

Output of this process is:

– modified reconstruction of the picture.

Let s’ be a variable specifying chroma sample array which is derived as follows.

– If cIdx is equal to 1, s’ represents the chroma sample array recPictureCb of the current picture.

– Otherwise (cIdx is equal to 2), s’ represents the chroma sample array recPictureCr of the current picture.

A variables β is derived as follows:

– If current slice is an intra coded slice, β is specified as with luma quantization parameter qPL as input.

– otherwise, β is set equal to 2\* qPL

A variable tC is specified as follows:

– If bS is greater than 2, the variable tC is specified specified as Table 8‑13 with luma quantization parameter ( qPL + 4 ) as input,

– Otherwise (bS is equal or less than 2), the variable tC is specified as Table 8‑13 with luma quantization parameter qPL as input.

Depending on verticalEdgeFlag, the following applies:

– If verticalEdgeFlag is equal to 1, for each sample location ( xC + xB, yC + yB + k ), k = 0..3, the following ordered steps apply:

1. The sample values pi and qi with i = 0..1 are derived as follows:

qi = s’[ xC + xB +i, yC + yB + k ] (8‑442)

pi = s’[ xC + xB – i – 1, yC + yB + k ] (8‑443)

1. If bS is greater than 1 and d is less than β, the following ordered steps apply:
2. The filtering process for a sample specified in subclause 8.6.1.4.6 is invoked with sample values pi, qi, with i = 0..1, the boundary filtering strength bS and the variable tC as inputs and the filtered sample values p0’ and q0’ as outputs.
3. The filtered sample values p0’ and q0’ replace the corresponding samples inside the sample array s’ as follows:

s’[ xC + xB , yC + yB + k ] = q0’ (8‑444)

s’[ xC + xB – 1, yC + yB + k ] = p0’ (8‑445)

– Otherwise (verticalEdgeFlag is equal to 0), the following ordered steps apply:

1. If xPOS is equal to 1, the parameters ks and ke are set to -1 and 2 respectively. If xD is equal to 2, the parameters ks and ke are set to 0 and 2 respectively. Otherwise ks and ke are set to 0 and 3 respectively.
2. The sample values pi and qi with i = 0..1 and k = ks..ke are derived as follows:

qi = s’[ xC + xB +k, yC + yB + i ] (8‑446)

pi = s’[ xC + xB +k, yC + yB – i – 1 ] (8‑447)

1. If xPOS is less than 0, and if bSL, is greater than 1 and d is less than β, for each sample location ( xC + xB - 1, yC + yB ), the following ordered steps apply:
2. The filtering process for a sample specified in subclause 8.6.1.4.6 is invoked with sample values pi, qi, with i = 0..1, the boundary filtering strength bSL and the variable tCL as inputs and the filtered sample values p0’ and q0’ as outputs.
3. The filtered sample values p0’ and q0’ replace the corresponding samples inside the sample array s’ as follows:

s’[ xC + xB +k, yC + yB ] = q0’ (8‑452)

s’[ xC + xB +k, yC + yB – i – 1 ] = pi’ (8‑445)

1. If bS is greater than 1 and d is less than β, for each sample location ( xC + xB + k, yC + yB ), k = 0.. ke, the following ordered steps apply:
2. The filtering process for a sample specified in subclause 8.6.1.4.6 is invoked with sample values pi, qi, with i = 0..1, the boundary filtering strength bS and the variable tC as inputs and the filtered sample values p0’ and q0’ as outputs.
3. The filtered sample values p0’ and q0’ replace the corresponding samples inside the sample array s’ as follows:

s’[ xC + xB +k, yC + yB ] = q0’ (8‑448)

s’[ xC + xB +k, yC + yB – 1 ] = p0’ (8‑449)

##### Decision process for a luma sample

[Ed: (WJ) no filtering when bS is equal to 0]

Inputs of this process are:

– sample values, pi and qi with i = 0..2,

– a variable bS specifying the boundary filtering strength,

– variables d, β and tC.

Output of this process is:

– a variable dSam containing a decision

When the variable bS is not equal to 0, the following applies:

– If d is less than ( β >> 2 ), | p3 – p0 | + | q0 – q3 | is less than ( β >> 3 ) and | p0 – q0 | is less than ( 5\*tC + 1 ) >> 1, dSam is set equal to 1.

– Otherwise, dSam is set equal to 0.

##### Filtering process for a luma sample

Inputs of this process are:

– sample values, pi and qi with i = 0..3,

– a variable dSam containing a decision,

– a variable tC.

Output of this process is:

– number of filtered samples nD,

– filtered sample values, pi’ and qi’ with i = 0..nD – 1,

Depending on dSam, the following applies:

– When the variable dSam is equal to 1, the following strong filtering applies while nD is set equal to 3:

p0’ = Clip1Y( ( p2 + 2\*p1 + 2\*p0 + 2\*q0 + q1 + 4 ) >> 3 ) (8‑450)

p1’ = Clip1Y( ( p2 + p1 + p0 + q0 + 2 ) >> 2 ) (8‑451)

p2’ = Clip1Y( ( 2\*p3 + 3\*p2 + p1 + p0 + q0 + 4 ) >> 3 ) (8‑452)

q0’ = Clip1Y( ( p1 + 2\*p0 + 2\*q0 + 2\*q1 + q2 + 4 ) >> 3 ) (8‑453)

q1’ = Clip1Y( ( p0 + q0 + q1 + q2 + 2 ) >> 2 ) (8‑454)

q2’ = Clip1Y( ( p0 + q0 + q1 + 3\*q2 + 2\*q3 + 4 ) >> 3 ) (8‑455)

– Otherwise, the following weak filtering applies while nD is set equal to 2:

Δ = Clip3( -tC, tC, ( 13\*( q0 – p0 ) + 4\*( q1 – p1 ) - 5\*( q2 – p0 ) + 16 ) >> 5 ) (8‑456)

p0’ = Clip1Y( p0 + Δ ) (8‑457)

q0’ = Clip1Y( q0 - Δ ) (8‑458)

p1’ = Clip1Y( p1 + Δ/2 ) (8‑459)

q1’ = Clip1Y( q1 – Δ/2 ) (8‑460)

Table ‑ – Derivation of threshold variables β and tC from input Q

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Q** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| **β** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 7 | 8 |
| **tC** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **Q** | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 |
| **β** | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 20 | 22 | 24 | 26 | 28 | 30 | 32 | 34 | 36 |
| **tC** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 4 | 4 | 4 |
| **Q** | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 |  |
| **β** | 38 | 40 | 42 | 44 | 46 | 48 | 50 | 52 | 54 | 56 | 58 | 60 | 62 | 64 | 64 | 64 | 64 | 64 |  |
| **tC** | 5 | 5 | 6 | 6 | 7 | 8 | 9 | 9 | 10 | 10 | 11 | 11 | 12 | 12 | 13 | 13 | 14 | 14 |  |

##### Filtering process for a chroma sample

[Ed: (WJ) no filtering when bS is equal or less than 2]

Inputs of this process are:

– sample values, pi and qi with i = 0..1,

– a variable bS specifying the boundary filtering strength.

– variable tC.

Output of this process is:

– The filtered sample values, p0’ and q0’.

When the variable bS is greater than 1, the filtered sample values p0’ and q0’ are derived by

Δ = Clip3( -tC, tC, ( ( 12 \* ( q0 – p0 )   +  3 \*(p1 – q1) + 16 ) >> 5 ) ) (8‑461)

p0’ = Clip1C( p0 + Δ ) (8‑462)

q0’ = Clip1C( q0 - Δ ) (8‑463)