**8.6.2.1.1 Modification process for luma and chroma samples**

Inputs to this process are:

– a variable cIdx specifying chroma component index,

– a sample position ( xC, yC ),

– a pair of variables ( rx, ry ) specifying the region index,

– a variable saoDepth specifying the split depth of the region,

– an array saoValueArray specifying offset values,

– a block size nS.

Output of this process is a modified picture buffer for the chroma component cIdx.

Let recSaoPicture represents the processed sample array of the current picture of chroma component cIdx and saoTypeIdx is set equal to sao\_type\_idx[ cIdx ][ saoDepth ][ rx ][ ry ].

Variable bitDepth is set equal to BitDepthY if cIdx is equal to 0, otherwise, set equal to BitDepthC.

Depending on the value of saoTypeIdx, the following applies:

– If saoTypeIdx is equal to one of the values of 1, 2, 3 or 4, the following ordered steps apply:

1. Arrays hPos[2] and vPos[2] are specified in Table 8‑16.
2. A variable edgeIdx is specified as

edgeIdx = 2 + ∑k( Sign( recPicture[ xC + i, yC + j ] –

recPicture[ xC + i + hPos[ k ], yC + j + vPos[ k ] ] ) ) with k = 0..1 (8‑463)

1. Each of the processed samples, recSaoPicture[ xC + i, yC + j ] with i = 0..nS-1 and j = 0..nS-1, is derived as follows:

– If pcm\_loop\_filter\_disable\_flag value is equal to 1 and the recPicture[ xC + i, yC + j ] is a reconstructed sample of an I\_PCM block, the recSaoPicture[ xC + i, yC + j ] is derived as

recSaoPicture[ xC + i, yC + j ] = recPicture[ xC + i, yC + j ] (8‑463)

– Otherwise, if **qpprime\_y\_zero\_transquant\_bypass\_flag is equal to 1 and** QP’Y is equal to 0, the recSaoPicture[ xC + i, yC + j ] is derived as

recSaoPicture[ xC + i, yC + j ] = recPicture[ xC + i, yC + j ] (8‑463)

– Otherwise (pcm\_loop\_filter\_disable\_flag value is equal to 0 or the recPicture[ xC + i, yC + j ] is not a reconstructed sample of an I\_PCM block), the recSaoPicture[ xC + i, yC + j ] is derived as

recSaoPicture[ xC + i, yC + j ] = recPicture[ xC + i, yC + j ] +

saoValueArray[ edgeTable[ edgeIdx ] ] (8‑463)

where edgeTable[5] = { 1, 2, 0, 3, 4}.

– Otherwise, if saoTypeIdx is equal to one of the values of 5 or 6, the following ordered steps applies:

1. A variable bandShift is set equal to BitDepthY – 5 if cIdx is equal to 0, otherwise, set equal to BitDepthC – 5.
2. Each of the processed samples, recSaoPicture[ xC + i, yC + j ] with i = 0..nS-1 and j = 0..nS-1, is derived as follows:

– If pcm\_loop\_filter\_disable\_flag value is equal to 1 and the recPicture[ xC + i, yC + j ] is a reconstructed sample of an I\_PCM block, the recSaoPicture[ xC + i, yC + j ] is derived as

recSaoPicture[ xC + i, yC + j ] = recPicture[ xC + i, yC + j ] (8‑463)

* Otherwise if **qpprime\_y\_zero\_transquant\_bypass\_flag is equal to 1 and** QP’Y is equal to 0, the recSaoPicture[ xC + i, yC + j ] is derived as

recSaoPicture[ xC + i, yC + j ] = recPicture[ xC + i, yC + j ] (8‑463)

– Otherwise (pcm\_loop\_filter\_disable\_flag value is equal to 0 or the recPicture[ xC + i, yC + j ] is not a reconstructed sample of an I\_PCM block), the recSaoPicture[ xC + i, yC + j ] is derived as

recSaoPicture[ xC + i, yC + j ] = recPicture[ xC + i, yC + j ] +

saoValueArray[ bandTable[ saoTypeIdx – 5 ][ bandIdx ] ] (8‑463)

where bandIdx is set equal to ( recPicture[ xC + i, yC + j ] >> bandShift ) and bandTable is specified in Table 8‑17.

– Otherwise (sao\_type\_idx[ cIdx ][ saoDepth ][ rx ][ ry ] is equal to 0), the following applies:

recSaoPicture[ xC + i, yC + j ] = recPicture[ xC + i, yC + j ] with i = 0..nS-1 and j = 0..nS-1 (8‑463)

[Ed. (WJ): copy operation is necessary to use recSaoPicture later.]