# CD changes

Proposed committee draft changes are described below using the following rules:

* Removed text from JCTVC-H1003 is shown with ~~strikethrough~~
* Added test is highlighted in yellow

**7.3.2.8 Sample adaptive offset VLC syntax**

|  |  |
| --- | --- |
| sao\_offset\_vlc( rx, ry, cIdx ) { | **Descriptor** |
| **sao\_type\_idx**[ cIdx ][ rx ][ ry ] | ue(v) |
| if( sao\_type\_idx[ cIdx ][ rx ][ ry ] = =5 ) { |  |
| **sao\_band\_position**[ cIdx ][ rx ][ ry ] | u(5) |
| for( i = 0; i < 4; i++ ) |  |
| **sao\_offset**[ cIdx ][ rx][ ry ][ i ] | se(v) |
| } else if( sao\_type\_idx[ cIdx ][ rx ][ ry ] != 0 ) |  |
| ~~for( i = 0; i < 4; i++ )~~ |  |
| **sao\_offset\_idx**[ cIdx ][ rx][ ry ]~~[ i ]~~ | u(4)~~ue(v)~~ |
| } |  |

#### Sample adaptive offset CABAC syntax

|  |  |
| --- | --- |
| sao\_offset\_cabac( rx, ry, cIdx ) { | Descriptor |
| **sao\_type\_idx**[ cIdx ][ rx ][ ry ] | ae(v) |
| if( sao\_type\_idx[ cIdx ][ rx ][ ry ] = =5 ) |  |
| **sao\_band\_position**[ cIdx ][ rx ][ ry ] | ae(v) |
| if( sao\_type\_idx[ cIdx ][ rx ][ ry ] != 0 ){ |  |
| for( i = 0; i < 4; i++ ) |  |
| **sao\_offset**[ cIdx ][ rx][ ry ][ i ] | ae(v) |
| }else |  |
| **sao\_offset\_idx**[ cIdx ][ rx][ ry ] | ae(v) |
| } |  |

* + - 1. **Sample adaptive offset VLC semantics**

**sao\_type\_idx**[ cIdx ][ rx ][ ry ] indicates the offset type as specified in Table 7‑6 of current coding treeblock at position rx and ry for the colour component cIdx.

When sao\_type\_idx[ cIdx ][ rx ][ ry ] is not present, it is inferred as follows.

* If sao\_merge\_up\_flag is equal to 1, sao\_type\_idx[ cIdx ][ rx ][ ry ] is set equal to sao\_type\_idx[ cIdx ][ rx ][ ry − 1 ].
* Otherwise, sao\_type\_idx[ cIdx ][ rx ][ ry ] is set equal to sao\_type\_idx[ cIdx ][ rx − 1 ][ ry ].

**Table 7‑6 – Specification of the edge type for SAO**

|  |  |
| --- | --- |
| **sao\_type\_idx[ cIdx ][ rx ][ ry ]** | **Edge type (informative)** |
| 0 | Not applied |
| 1 | 1D 0-degree edge |
| 2 | 1D 90-degree edge |
| 3 | 1D 135-degree edge |
| 4 | 1D 45-degree edge |
| 5 | Band |

**sao\_band\_position**[ cIdx ][ rx ][ ry ] indicates the displacement of the band offset of the pixel range when sao\_type\_idx[ cIdx ][ rx ][ ry ] is equal to 5.

When sao\_band\_position[ cIdx ][ rx ][ ry ] is not present it is inferred as follows.

* If sao\_merge\_up\_flag is equal to 1, sao\_band\_position[ cIdx ][ rx ][ ry ] is set equal to sao\_band\_position[ cIdx ][ rx ][ ry − 1 ].
* Otherwise, sao\_band\_position[ cIdx ][ rx ][ ry ] is set equal to sao\_band\_position[ cIdx ][ rx − 1 ][ ry ].

**sao\_offset**[ cIdx ][ rx ][ ry ][ i ] indicates the offset value of i-th category of current coding treeblock at position rx and ry for the colour component cIdx.

**sao\_offset\_idx**[ cIdx ][ rx ][ ry ] indicates the index of the group of offsets for each category whenEdge type is equal to 1, 2, 3 or 4. The group of offsets is derived from the value of sao\_offset\_idx[ cIdx ][ rx ][ ry ].

The variable bitDepth is derived as follows.

* If cIdx is equal to 0, bitDepth is set equal to BitDepthY..
* Otherwise (cIdx is equal to1 or 2), bitDepth is set equal to BitDepthC.

It is a requirement of bitstream conformance that when sao\_type\_idx[ cIdx ][ rx ][ ry ] is not equal to 5, the values of sao\_offset[ cIdx ][ rx ][ ry ][ i ] shall be in the range of 0 to ( 1<< ( Min( bitDepth, 10 ) − 5 ) ) − 1, inclusive and otherwise, the values of sao\_offset[ cIdx ][ rx ][ ry ][ i ] shall be in the range of −( 1<< ( Min( bitDepth, 10 ) − 5 ) ) to ( 1<< ( Min( bitDepth, 10 ) − 5 ) ) − 1, inclusive.

When sao\_offset[ cIdx ][ rx ][ ry ][ i ] is not present, it is inferred as follows.

* If sao\_merge\_up\_flag is equal to 1, sao\_offset[ cIdx ][ rx ][ ry ][ i ] is set equal to sao\_offset[ cIdx ][ rx ][ ry − 1 ][ i ].
* Otherwise, sao\_offset[ cIdx ][ rx ][ ry ][ i ] is set equal to sao\_offset[ cIdx ][ rx − 1 ][ ry ][ i ].

~~The variable offsetSign is derived as follows.~~

* ~~If sao\_type\_idx[ cIdx ][ rx ][ ry ] is less than 5 and i is larger than 1,offsetSign is set to −1.~~
* ~~Otherwise (sao\_type\_idx[ cIdx ][ rx ][ ry ] is equal to 5 or i is less than 2), offsetSign is set to 1.~~

The array SaoOffsetVal is derived as follows.

SaoOffsetVal[ cIdx ][ rx ][ ry ][ 0 ] = 0 (7‑36)

~~SaoOffsetVal[ cIdx ][ rx ][ ry ][ i + 1 ] =   
 offsetSign\*sao\_offset[ cIdx ][ rx ][ ry ][ i ] << ( bitDepth – Min( bitDepth, 10 ) )~~ ~~(7‑37)~~

SaoOffsetVal[ cIdx ][ rx ][ ry ][ 1 ] = Round ((**sao\_offset\_idx**[ cIdx ][ rx ][ ry ] + 1)3) (7‑36)

SaoOffsetVal[ cIdx ][ rx ][ ry ][ 2 ] = Round (**sao\_offset\_idx**[ cIdx ][ rx ][ ry ]/3)%2 (7‑36)

SaoOffsetVal[ cIdx ][ rx ][ ry ][ 3 ] = - (**sao\_offset\_idx**[ cIdx ][ rx ][ ry ]+1)%2 (7‑36)

SaoOffsetVal[ cIdx ][ rx ][ ry ][ 4 ] = - SaoOffsetVal[ cIdx ][ rx ][ ry ][ 1 ]

+ SaoOffsetVal[ cIdx ][ rx ][ ry ][ 2 ] + SaoOffsetVal[ cIdx ][ rx ][ ry ][ 3 ] (7‑36)

| Table 9‑32 – Syntax elements and associated types of binarization, maxBinIdxCtx, ctxIdxTable, and ctxIdxOffset | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Syntax element** | **initialisationType** | **Type of binarization** | **maxBinIdxCtx** | **ctxIdxTable** | **ctxIdxOffset** |
| sao\_offset\_idx | 0 | FL, cMax = 4 | na | na | na, (uses Decode Bypass) |
| 1 | na | na | na, (uses Decode Bypass) |
| 2 | na | na | na, (uses Decode Bypass) |