|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  20th Meeting: Geneva, CH, 10–18 Feb. 2015 | Document: JCTVC-T0118 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Non-CE1: On escape color coding for palette coding mode** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Xiaoyu Xiu, Yan Ye, Yuwen He 9710 Scranton Rd, #250 San Diego, CA 92121, USA | Tel: Email: | + 1-858-210-4830 [Xiaoyu.Xiu@InterDigital.com](mailto:Xiaoyu.Xiu@InterDigital.com)  [Yan.Ye@InterDigital.com](mailto:Yan.Ye@InterDigital.com)  Yuwen.He@InterDigital.com |
| *Source:* | InterDigital Communications, LLC | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

This contribution proposes to improve the escape color coding method for the existing palette design in HEVC screen content coding specification draft 2 and the test model SCM-3.0. Specifically, two defects of the current escape color coding method are identified. Firstly, the calculation of the maximum value for the truncated binary code of escape colors doesn’t match the actual dynamic range of quantized escape colors. This affects the efficiency of palette coding mode, especially for medium to low QPs, due to the insufficient dynamic range represented by the truncated binary codewords. Secondly, the inverse quantization process of escape colors is not properly defined, causing the right-shifts to become negative for high QPs. Solutions are proposed to resolve the above issues in the existing design, and tested for both low QPs (*i.e*., 0, 1, 2, 3) and high QPs (*i.e.*, 42, 43, 44, 45) in addition to the QP settings in the common test conditions.

Compared to SCM-3.0 lossy anchor, for low QPs, the proposed methods provide the average {G/Y, B/Cb, R/Cr} BD-rate savings for AI, RA and LB of {13%, 10%, 10%}, {6.5%, 5.7%, 5.6%} and {2.1%, 2.1%, 2.1%}, respectively, for the sequences in the categories *text & graphics with motion, 1080p & 720p* in both RGB and YCbCr color formats. For high QPs, the corresponding average {G/Y, B/Cb, R/Cr} BD-rate savings are {0.1%, 0.2%, 0.1%}, {0.2%, 0.2%, 0.1%} and {0.0%, 1.4%, 0.2%} respectively. Additionally, the proposed methods do not bring any performance loss for the QP settings in the common test conditions.

# Problem statement

Based on the current design of palette coding mode, for each pixel coded as escape color, its color value needs to be transmitted to decoder. Additionally, when lossy coding is applied, the values of escape colors will be quantized before being signaled into bit-stream. To reduce the overhead of escape color signaling, the quantized (if lossy coding is applied) escape colors are binarized using truncated binary code (TBC) and coded in bypass mode. TBC is a variant of fixed length code (FLC) which needs the possible maximum level to be specified as input to the corresponding binarization process.

## The maximum TBC level for escape colors

According to the SCM-3.0, for lossy coding, the maximum TBC level for escape colors is derived as follows:

1. The rounded quantization step size is derived as

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

1. A quantized value is calculated based on the maximum escape color value and the rounded quantization step size , as follows

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

1. The number of bits, , to represent is derived as

|  |  |  |
| --- | --- | --- |
|  | *numBins = 0*  *While(maxValue) {*  *maxValue = maxValue >>1*  *numBins ++ }* | (3) |

1. The maximum quantized value that could be possibly achieved is derived by

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

where contains the scaling factors used for quantization process, i.e., [26214, 23302, 20560, 18396, 16384, 14564].

1. The maximum TBC value is finally obtained by

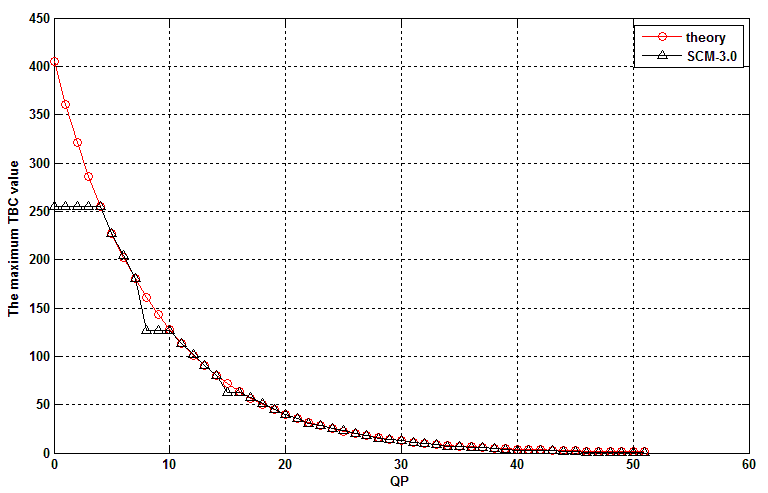
|  |  |  |
| --- | --- | --- |
|  |  | (5) |

As shown above, in Step 1, the rounded quantization step size is calculated by rounding the real-valued quantization step size to the nearest integer value (as shown in (1)). Such rounding of is very coarse, and introduces significant loss of precision into the division in Step 2. This loss of precision is more serious for small QPs. As a consequence, the value thus calculated could be smaller than the actual maximum quantized value of escape colors, due to the clipping applied in step 5. Then, when the quantized escape color value is clipped using , the reconstructed quality of palette-coded CUs is compromised.

To illustrate this issue, assuming the input bit-depth is 8-bit (i.e., ), Figure 1 depicts the calculated maximum TBC levels for escape colors at different QPs, based on the above steps. To facilitate understanding, an additional curve is also added, which represents the theoretical maximum value of the quantized escape colors as depicted as follows, assuming floating point operations:

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

As we can see in Figure 1, the current calculation method leads to smaller maximum TBC values (by up to -149.79 when QP is equal to 0) compared to the actual theoretical results. The differences are much larger at small QPs, because the rounding errors of introduced by the coarse rounding of step 1 are more severe at small QPs. Furthermore, the existing calculation of the TBC maximum level involves floating-point operation in (1), which is not preferred for practical hardware implementations.



**Figure 1 The maximum TBC level for escape colors at different QPs.**

## The inverse quantization of escape colors

It was found that the escape color de-quantization process in HEVC SCC specification draft 2 [1] is inconsistent with SCM-3.0, and the former is incorrectly specified. Therefore, we use the inverse quantization process defined in SCM-3.0 for the following discussion. In SCM-3.0, the inverse quantization of escape colors is defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

where is the value of quantized escape color and specifies the scaling factors used for the inverse quantization process. As shown in (2), the right shift of bits is only properly defined when . When , the corresponding right shift becomes negative, for which the corresponding operation is undefined.

# Proposed methods

In this section, several solutions are proposed to address the problems detailed above for escape color coding. Specifically, the proposed solutions can be summarized as

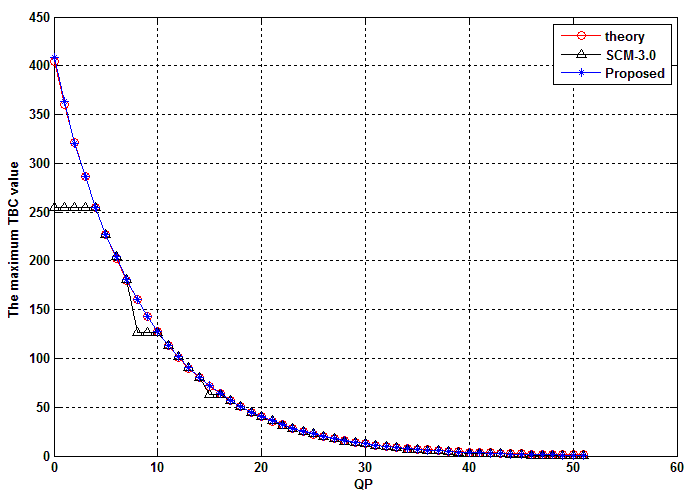
1. Improved maximum TBC level calculation method for escape colors to ensure that the TBC codewords can cover the actual dynamic range of quantized escape colors.
2. Fixed inverse quantization process for escape colors to ensure that the problem of negative right shift will not occur for all the QPs.

## The proposed solution to derive the maximum TBC level for escape colors

We propose to fix the calculation of maximum TBC level for escape colors such that it covers the actual dynamic range of quantized escape colors. Specifically, in the proposed method, the maximum TBC value for escape colors is derived according to the quantization process using the same method as the calculation of the maximum quantized value in (4), as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

where specifies the scaling factors used for quantization process, i.e., . Figure 2 shows the corresponding maximum TBC levels for escape colors at different QPs using the proposed method. As shown in Figure 2, it can be observed that the TBC maximum levels calculated by the proposed method are very close to the theoretical values.

****

**Figure 2 The maximum TBC level for escape color signaling at different QPs using the proposed method.**

## Bit-stream conformance constraint on escape color signaling

Because the TBC codewords have limited range that does not exceed the maximum TBC level, it is proposed to add bit-stream conformance requirement to require that the signaled values of escape colors (as represented by the syntax element *palette\_escape\_val*) cannot be larger than the maximum level of the TBC code, i.e., .

## The proposed solutions for the inverse quantization of escape colors

As shown in (7), the right shift of bits is only defined for . When , the right shift becomes a negative value, for which the corresponding operation is undefined. To resolve this problem, it is proposed to move partial right shifts that are related with the input QP value to the scaling operation of the inverse quantization process, that is, right shift of –() bits becomes left shift of () bits. The inverse quantization process to generate the reconstructed escape colors can be written as

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

# Simulation results

The proposed methods are implemented based on SCM-3.0. Both the proposed methods on the maximum TBC level for escape colors in Section 2.1 and the inverse quantization of escape colors in Section 2.3 are tested using the common test condition (CTC) as specified in [2]. As analyzed previously, for the current design of escape color coding, the problem of insufficient dynamic range of the TBC codewords (as specified in Section 1.1) is more severe for low QPs, whereas the problem of negative right shift for inverse quantization (as specified in Section 1.2) only occurs for high QPs (i.e., ). Therefore, in addition to the CTC, the proposed methods are also compared with SCM-3.0 anchor for a set of low QPs {0, 1, 2, 3} and a set of high QPs {42, 43, 44, 45}.

## The comparison for the QPs of the current CTC

Table 1 present the BD-rate performance of the proposed method compared to the SCM-3.0 anchor under CTC. It can be seen that the proposed method do not bring any performance difference compared to the anchors.

**Table 1 BD-rate performance of the proposed method, compared to SCM-3.0 anchor (QP = 22, 27, 32, 37)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **All Intra** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | 0.0% | 0.0% | 0.0% |
| RGB, mixed content, 1440p & 1080p | 0.0% | 0.0% | 0.0% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | 0.0% | 0.0% | 0.0% |
| YUV, mixed content, 1440p & 1080p | 0.0% | 0.0% | 0.0% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 93% | | |
| Dec Time[%] | 94% | | |
|  |  |  |  |
|  | **Random Access** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | 0.0% | 0.0% | 0.0% |
| RGB, mixed content, 1440p & 1080p | 0.0% | 0.0% | 0.0% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | 0.0% | 0.0% | 0.0% |
| YUV, mixed content, 1440p & 1080p | 0.0% | 0.0% | 0.0% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 97% | | |
| Dec Time[%] | 94% | | |
|  |  |  |  |
|  | **Low delay B** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | 0.0% | 0.0% | 0.0% |
| RGB, mixed content, 1440p & 1080p | 0.0% | 0.0% | 0.0% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | 0.0% | 0.0% | 0.0% |
| YUV, mixed content, 1440p & 1080p | 0.0% | 0.0% | 0.0% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 96% | | |
| Dec Time[%] | 97% | | |

## The comparison for the low QPs

Using the performance of SCM-3.0 as anchor, Table 2 illustrates the BD-rate saving of the proposed method for the low QPs. The unusual BD-rate performance for the category *YUV*, *text & graphics with motion, 1080p & 720p* for AI configuration was caused by the abnormal BD-rate performance of the SCM-3.0 anchor for the sequence *sc\_console\_1920x1080\_60\_8bit\_444* (as shown in Figure 3). Although the abnormal behavior is fixed with the proposed solution, the BD-rate calculation method cannot properly reflect that. Excluding this problem sequence, the proposed method provides the average {G/Y, B/Cb, R/Cr} BD-rate savings for AI, RA and LB of {13%, 10%, 10%}, {6.5%, 5.7%, 5.6%} and {2.1%, 2.1%, 2.1%}, respectively, for the sequences in the categories *text & graphics with motion, 1080p & 720p* in both RGB and YCbCr color formats.

**Table 2 BD-rate performance of the proposed method, compared to SCM-3.0 anchor (QP = 0, 1, 2, 3)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **All Intra** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | -15.3% | -13.0% | -12.8% |
| RGB, mixed content, 1440p & 1080p | -2.2% | -2.0% | -2.0% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | 170.4% | -8.3% | -8.5% |
| YUV, mixed content, 1440p & 1080p | -1.1% | -0.9% | -0.9% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 98% | | |
| Dec Time[%] | 95% | | |
|  |  |  |  |
|  | **Random Access** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | -7.6% | -7.7% | -7.6% |
| RGB, mixed content, 1440p & 1080p | -0.7% | -0.7% | -0.7% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | -5.4% | -3.7% | -3.6% |
| YUV, mixed content, 1440p & 1080p | -0.2% | -0.2% | -0.2% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 99% | | |
| Dec Time[%] | 98% | | |
|  |  |  |  |
|  | **Low delay B** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | -2.7% | -2.8% | -2.8% |
| RGB, mixed content, 1440p & 1080p | -0.3% | -0.2% | -0.3% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | -1.5% | -1.3% | -1.3% |
| YUV, mixed content, 1440p & 1080p | -0.1% | -0.1% | -0.1% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 98% | | |
| Dec Time[%] | 96% | | |

**Figure 3 The BD-rate performances of *sc\_console\_1920x1080\_60\_8bit\_444* for AI configuration at low QPs.**

## The comparison for the high QPs

Using the performance of SCM-3.0 as anchor, Table 3 illustrates the BD-rate saving of the proposed method for the high QPs. It can be seen that for high QPs the proposed method provides the average {G/Y, B/Cb, R/Cr} BD-rate savings for AI, RA and LB of {0.1%, 0.2%, 0.1%}, {0.2%, 0.2%, 0.1%} and {0.0%, 1.4%, 0.2%}, respectively, for the sequences in the categories *text & graphics with motion, 1080p & 720p* in both RGB and YCbCr color formats. It is worth noting that at high QPs, the number of escape colors is very small; therefore the average BD-rate savings also appeared to be small, regardless of the significant impact on the escape colors due to the incorrect behavior. In fact, it is observed that the negative right shift could cause incoherent behaviors depending on the simulation platform used, and consequently cause the reconstructed escape colors to be significantly distorted.

**Table 3 BD-rate performance of the proposed method, compared to SCM-3.0 anchor (QP = 42, 43, 44, 45)**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **All Intra** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | -0.3% | -0.4% | -0.5% |
| RGB, mixed content, 1440p & 1080p | -0.1% | -0.4% | -0.5% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | 0.0% | -0.1% | 0.3% |
| YUV, mixed content, 1440p & 1080p | 0.0% | 0.0% | 0.0% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 98% | | |
| Dec Time[%] | 96% | | |
|  |  |  |  |
|  | **Random Access** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | -0.2% | -0.3% | -0.3% |
| RGB, mixed content, 1440p & 1080p | -0.1% | -0.5% | -0.4% |
| RGB, Animation, 720p | 0.0% | 0.1% | 0.1% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | -0.1% | 0.0% | 0.0% |
| YUV, mixed content, 1440p & 1080p | 0.1% | 0.2% | 0.4% |
| YUV, Animation, 720p | 0.0% | 0.1% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 100% | | |
| Dec Time[%] | 96% | | |
|  |  |  |  |
|  | **Low delay B** | | |
|  | G/Y | B/U | R/V |
| RGB, text & graphics with motion, 1080p & 720p | -0.2% | -2.7% | -0.3% |
| RGB, mixed content, 1440p & 1080p | 0.2% | -0.1% | -0.3% |
| RGB, Animation, 720p | 0.0% | 0.0% | 0.0% |
| RGB, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| YUV, text & graphics with motion, 1080p & 720p | 0.1% | -0.1% | -0.1% |
| YUV, mixed content, 1440p & 1080p | -0.2% | 0.3% | 1.1% |
| YUV, Animation, 720p | 0.0% | 0.0% | 0.0% |
| YUV, camera captured, 1080p | 0.0% | 0.0% | 0.0% |
| Enc Time[%] | 99% | | |
| Dec Time[%] | 95% | | |
|  |  |  |  |

# Proposed specification changes

#### 7.4.9.6 Palette mode semantics

**palette\_escape\_val** specifies the quantized escape coded sample value for a component.

It is a requirement of bitstream conformance for the input bitstream that the value of palette\_escape\_val shall be equal to or smaller than the maximum parameter of the TB binarization process specified in 9.3.3.13.

The variable PaletteEscapeVal**[** cIdx **][** xC **][** yC **]** specifies escape value of a sample for which paletteSampleMode[ xC ][ yC ] is equal to ESCAPE\_MODE. The array index cIdx specifies the colour component. The array indices xC, yC specify the location ( xC, yC ) of the sample relative to the top-left luma sample of the picture.

#### 8.4.5.2.8 Decoding process for palette mode

……

The (nCbS x nCbS) block of the reconstructed sample array recSamples at location ( xCb, yCb ) is derived as follows:

For x = 0..nCbS − 1,  y = 0..nCbS − 1, recSample[ cIdx ][ yCb + y ][ xCb + x ] is set as follows:

– If paletteSampleMode[ xCb + x ][ yCb + y ] is not equal to ESCAPE\_MODE, the following applies:

recSample[ cIdx ][ xCb + x ][ yCb + y ] = currentPaletteEntries[ cIdx ][ paletteIndexMap[ xCb + x ][ yCb + y ] ],

– Otherwise, if cu\_transquant\_bypass\_flag is true, the following applies:

recSample[ cIdx ][ xCb + x ][ yCb + y ] = paletteEscapeVal[ cIdx ][ xCb + x ][ yCb + y ],

– Otherwise (paletteSampleMode[ xCb + x ][ yCb + y ] is equal to ESCAPE\_MODE and cu\_transquant\_bypass\_flag is false), the following ordered steps apply:

1. The derivation process for quantization parameters as specified in subclause 8.6.1 is invoked with the location ( xCb, yCb ) specifying the top-left sample of the current block relative to the top-left sample of the current picture.
2. The quantization parameter qP is derived as follows:

– If cIdx is equal to 0,

qP = max( 0, Qp′Y ) (8‑XX)

– Otherwise, if cIdx is equal to 1,

qP = max( 0, Qp′Cb ) (8‑XX)

– Otherwise (cIdx is equal to 2),

qP = max(0, Qp′Cr ) (8‑XX)

1. The variables bitDepth ~~and bdShift1 are~~ is derived as follows:

bitDepth = ( cIdx = = 0 ) ? BitDepthY : BitDepthC  (8‑XX)

~~bdShift1 = Max( 20 − bitDepth, extended\_precision\_processing\_flag ? 11 : 0 ) (8‑XX)~~

1. ~~The variables log2TransformRange, bdShift2, coeffMin and coeffMax are derived as follows:~~

~~– If cIdx is equal to 0,~~

~~log2TransformRange = extended\_precision\_processing\_flag ? Max( 15, BitDepth~~~~Y~~ ~~+ 6 ) : 15 (8‑XX)~~

~~bdShift2 = BitDepth~~~~Y~~~~+ Log2( nTbS ) + 10 − log2TransformRange (8‑XX)~~

~~coeffMin = CoeffMin~~~~Y~~ ~~(8‑XX)~~

~~coeffMax = CoeffMax~~~~Y~~ ~~(8‑XX)~~

~~– Otherwise,~~

~~log2TransformRange = extended\_precision\_processing\_flag ? Max( 15, BitDepth~~~~C~~ ~~+ 6 ) : 15 (8‑XX)~~

~~bdShift2 = BitDepth~~~~C~~~~+ Log2( nTbS ) + 10 − log2TransformRange (8‑XX)~~

~~coeffMin = CoeffMin~~~~C~~ ~~(8‑XX)~~

~~coeffMax = CoeffMax~~~~C~~ ~~(8‑XX)~~

1. The list levelScale[ ] is specified as levelScale[ k ] = { 40, 45, 51, 57, 64, 72 } with k = 0..5.
2. The following applies

~~recSamples[ xCb + x ][ yCb + y ] =  
Clip3( coeffMin, coeffMax, ( ( palette\_escape\_val[ cIdx ][ xCb + x ][ yCb + y ]\* 16 \*   
 levelScale[ qP%6 ]  <<  (qP / 6 ) ) + ( 1  <<  ( bdShift2 − 1 ) ) )  >>  bdShift2 ) (8‑XX)~~

recSamples[ cIdx ] [ xCb + x ][ yCb + y ] =  
( ( palette\_escape\_val[ cIdx ][ xCb + x ][ yCb + y ] \*   
 levelScale[ qP%6 ] ) <<  (qP / 6 ) + 32 )  >>  6 (8‑XX)

1. ~~The reconstructed sample value recSamples[ xCb + x ][ yCb + y ] is modified as follows:~~

~~recSamples[ xCb + x ][ yCb + y ] = (recSamples[ xCb + x ][ yCb + y ]   
 + ( 1  <<  ( bdShift2 − 1 ) ) ) >> bdShift2 (8‑XX)~~

#### 9.3.3.13 Binarization process for palette\_escape\_val

Input to this process is a request for a binarization for the syntax element palette\_escape\_val, cu\_transquant\_bypass\_flag and colour component index cIdx.

Output of this process is the binarization of palette\_escape\_val.

The variable bitDepth is derived as follows:

bitDepth = ( cIdx = = 0 ) ? BitDepthY : BitDepthC

The binarization of palette\_escape\_val is derived as follows:

* If cu\_transquant\_bypass\_flag is true, the binarization of palette\_escape\_val is derived by invoking the FL binarization process specified in subclause 9.3.3.5 with the input parameter set to (1<<bitdepth) − 1.
* Otherwise (cu\_transquant\_bypass\_flag is false) the following ordered steps apply

1. The quantization parameter qP is derived as follows:

qP = ( cIdx = = 0 ) ? Qp′Y : ( ( cIdx = = 1 ) Qp′Cb ? Qp′Cr ) (9-XX)

1. ~~A quantization step size parameter qStep is derived as follows:~~

~~qStep = ( qP= =0 ) ? Round( 2~~~~(qP − 4) / 6~~~~) : 1~~

1. ~~A maximum possible quantized value maxValue is derived as follows:~~

~~maxValue = ( ( 1<<bitDepth ) − 1 ) / Qstep~~

1. ~~The number of bins numBins of the fixed length binarization codeword is derived as follows~~

~~while( maxValue ) {  
 maxValue = maxValue >> 1  
 numBins++  
 }~~

1. ~~The maximum parameter cMax for the fixed length binarization is derived as follows~~

~~cMax = ( 1 << numBins ) – 1~~

1. The variable bdShift is derived as follows:

bdShift = 14 + qP/6 (9-XX)

1. The list quantScale[ ] is specified as quantScale[ k ] = [26214, 23302, 20560, 18396, 16384, 14564] with k = 0, 1, 2, …, 5.
2. The maximum parameter cMax for TB binarization process is derived as follows:

cMax = ( ( ( 1<< bitDepth ) − 1 ) \*  quantScale[ qP%6 ] + ( 1 << ( bdShift - 1) ) ) >> bdShift (9-XX)

The binarization for the palette\_escape\_value is derived by invoking the TB binarization process specified in subclause 9.3.3.6 with cMax.

# References

1. R. Joshi, J. Xu, HEVC Screen Content Coding Draft Text 2, JCTVC-S1005, Oct. 2014, Strasbourg, France.
2. H. Yu, R. Cohen, K. Rapaka, J. Xu, Common Test Conditions for Screen Content Coding, JCTVC-S1015, Oct. 2014, Strasbourg, France.

# Patent rights declaration(s)

**InterDigital Communications LLC may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard (per box 2 of the ITU-T/ITU-R/ISO/IEC patent statement and licensing declaration form).**