



# JVET-N0220: AHG16: Simplification of LMCS (Reshaper) Implementation

---

Mar. 2019

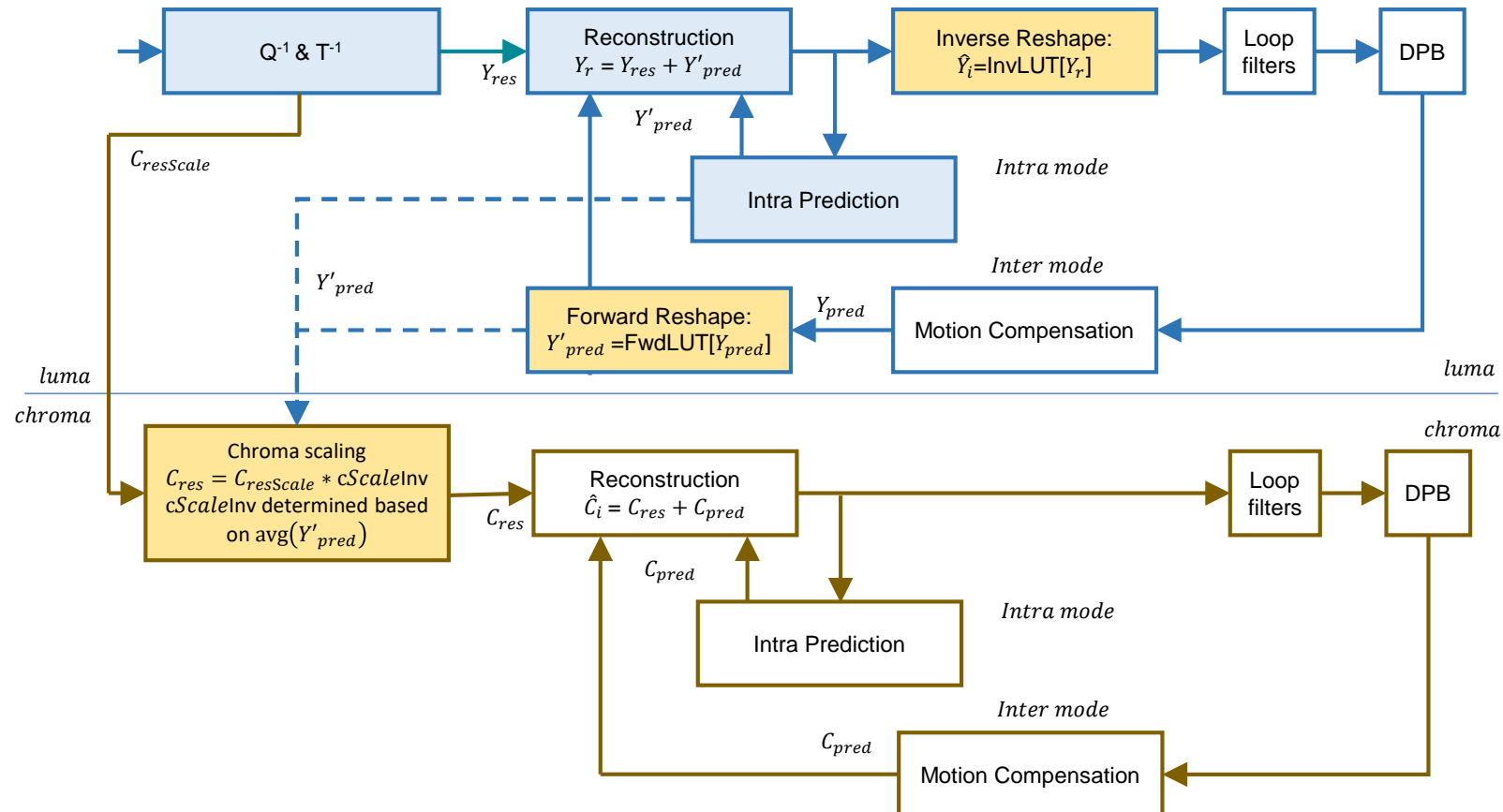
Dolby Laboratories

# Overview

- Introduction
- Simplification of LMCS (reshaper) implementation (normative)
  - Reduced pipeline delay
  - Reduced local buffer size
  - Unified luma and chroma fixed-point precision
  - Unified luma and chroma scaling factor calculation method
- Non-CTC non-normative topics (informative)
  - VTM software modification for other bit depths
  - Interaction of LMCS and local QP adjustment
  - SCC TGM results
- Conclusion

# Introduction

- The LMCS added in VTM 4.0:
  - in-loop mapping of the luma component based on adaptive piecewise linear models
  - luma-dependent chroma residual scaling for the chroma components

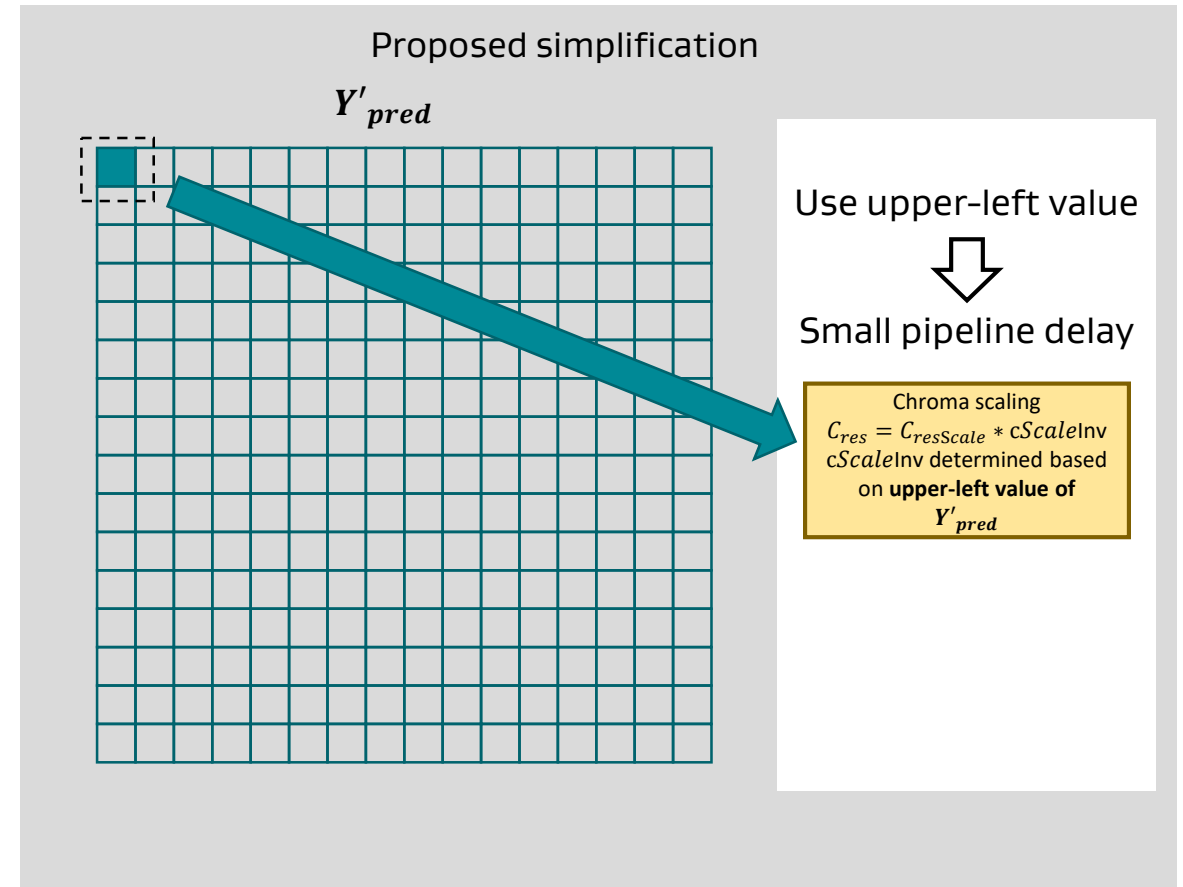
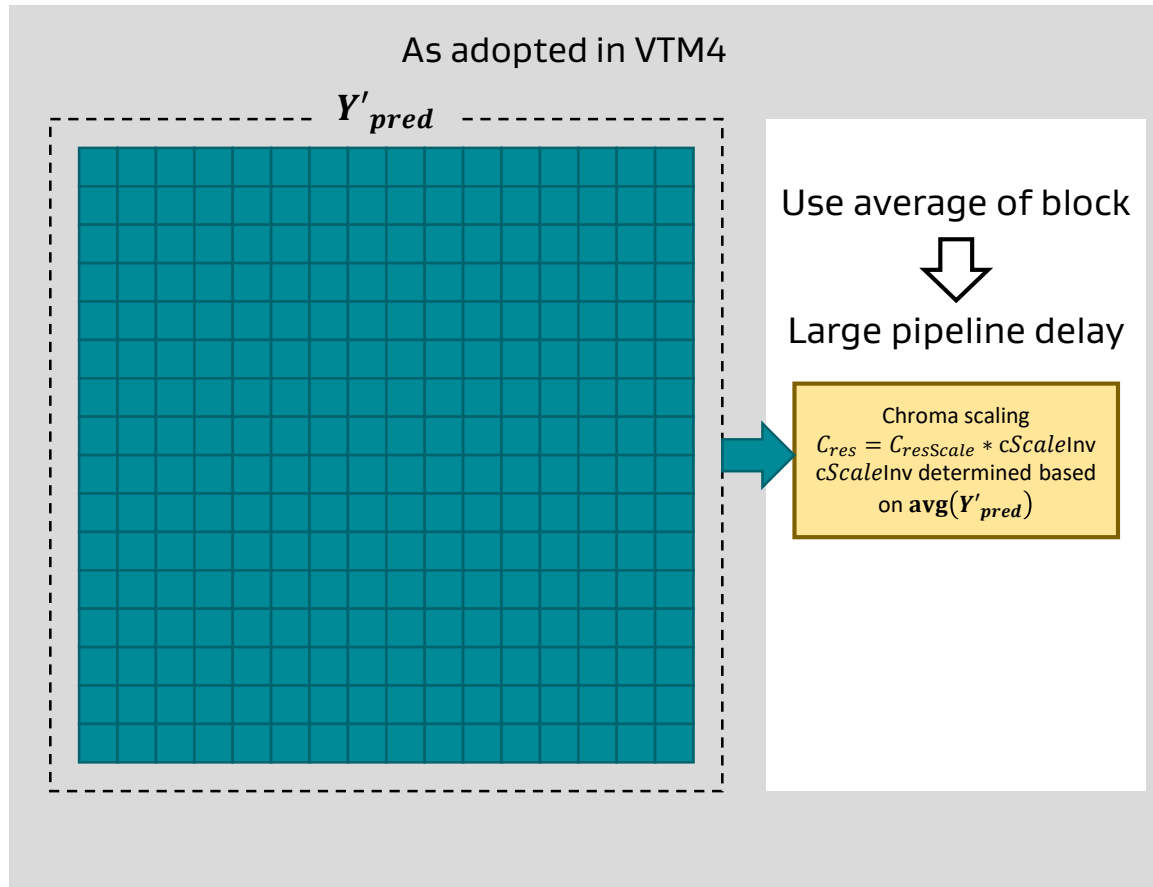


# Overview

- Introduction
- Simplification of LMCS implementation (normative)
  - Reduced pipeline delay
  - Reduced local buffer
  - Unified luma and chroma fixed-point precision
  - Unified luma and chroma scaling factor calculation method
- Non-CTC non-normative topics (informative)
  - VTM modification for other bit depths
  - Interaction of LMCS and local QP adjustment
  - SCC TGM results
- Conclusion

# Simplification 1: Reduction of pipeline delay

Tests show reasonably good linear correlation between  $\text{avg}(Y'_{pred})$  and **upper-left value of  $Y'_{pred}$**



# Results - Simplification 1: Reduction of pipeline delay

**Simplification 1 only – Test results on SDR CTC over VTM4.0 anchors**

	All Intra Main10			Random access Main10			Low delay B Main10			Low delay P Main10		
	Y	U	V	Y	U	V	Y	U	V	Y	U	V
Class A1	0.00%	0.00%	0.00%	0.00%	-0.07%	0.00%						
Class A2	0.00%	0.00%	0.00%	-0.03%	-0.05%	-0.03%						
Class B	0.00%	0.00%	0.00%	0.01%	-0.06%	-0.02%	0.00%	-0.07%	0.08%	0.00%	-0.14%	0.16%
Class C	0.00%	0.00%	0.00%	0.02%	0.02%	-0.18%	-0.03%	0.46%	-0.07%	0.01%	-0.54%	-0.30%
Class E	0.00%	0.00%	0.00%				-0.05%	0.01%	-0.63%	0.00%	-0.26%	0.00%
<b>Overall</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>-0.04%</b>	<b>-0.06%</b>	<b>-0.02%</b>	<b>0.13%</b>	<b>-0.15%</b>	<b>0.00%</b>	<b>-0.30%</b>	<b>-0.03%</b>
Class D	0.00%	0.00%	0.00%	0.01%	-0.03%	0.15%	-0.03%	-0.41%	-0.71%	0.05%	-0.20%	-0.02%
Class F	0.00%	0.00%	0.00%	-0.01%	0.05%	0.00%	-0.16%	-0.36%	0.00%	0.01%	0.15%	0.18%

Change in overall BD-rate is very small

Enc/Dec time is similar as VTM4.0

# Simplification 2: Reduction of local buffer size

## As adopted in VTM4

For hardware implementation, chroma residue needs to be kept in high-precision before it is scaled (e.g., from 11-bit storage to 16-bit storage for 10-bit video).

It costs approximately 2.5 Kbytes additional local buffer size (double buffer, i.e.  $5\text{-bit} * 32 * 32 * 2 * 2/8 = 2560$  bytes) for a 64x64 decoder pipeline.

## Proposed simplification

Add a **normative clipping** in the chroma residue scaling process.

For each chroma residue sample to be scaled, clip it within the valid residue range given the bitdepth of the chroma components before chroma residue scaling:

$$CRes = \text{Clip3}(- (1 \ll \text{BitDepthC}), 1 \ll \text{BitDepthC} - 1, CRes)$$



# Results - Simplification 2: Reduction of local buffer size

## Simplification 2 only – Test results on SDR CTC over VTM4.0 anchors

	All Intra Main10			Random access Main10			Low delay B Main10			Low delay P Main10		
	Y	U	V	Y	U	V	Y	U	V	Y	U	V
Class A1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Class A2	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%						
Class B	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%						
Class C	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%						
Class E	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%						
<b>Overall</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>
Class D	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Class F	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

There is no measurable change in BD-rate

Enc/Dec time is similar as VTM4.0



# Simplifications 3 and 4: Unification of calculations

Simplification 3: Unification of fixed-point precision for luma and chroma scaling

Simplification 4: Unification of calculation of luma and chroma scaling factor

As adopted in VTM4

```
shiftY = 14
ReshapePivot[ 0 ] = 0;
for( i = 0; i <= MaxBinIdx ; i++) {
    ReshapePivot[ i + 1 ] = ReshapePivot[ i ] + rspCW[ i ]
    ScaleCoeff[ i ] = ( rspCW[ i ] * (1 << shiftY) + (1 << (Log2(OrgCW) - 1))) >> (Log2(OrgCW))
    if ( RspCW[ i ] == 0 )
        InvScaleCoeff[ i ] = 0
    else
        InvScaleCoeff[ i ] = OrgCW * (1 << shiftY) / rspCW[ i ]
}
```

The variable ChromaScaleCoeff[ i ] with i in the range of 0 to MaxBinIdx , inclusive, are derived as follows:  
chromaResidualScaleLut[64] = { 16384, 16384, 16384, 16384, 16384, 16384, 16384, 8192, 8192, 8192, 8192, 5461, 5461, 5461, 5461, 4096, 4096, 4096, 4096, 3277, 3277, 3277, 3277, 2731, 2731, 2731, 2731, 2341, 2341, 2341, 2048, 2048, 2048, 1820, 1820, 1820, 1638, 1638, 1638, 1638, 1489, 1489, 1489, 1489, 1365, 1365, 1365, 1365, 1260, 1260, 1260, 1260, 1170, 1170, 1170, 1170, 1092, 1092, 1092, 1092, 1024, 1024, 1024, 1024};

```
shiftC = 11
- if ( RspCW[ i ] == 0 )

    ChromaScaleCoef [ i ] = (1 << shiftC)
- Otherwise (RspCW[ i ] != 0), the following applies:
binCW = BitDepthY > 10 ? ( rspCW[ i ] >> (BitDepthY - 10)) : BitDepthY < 10 ? ( rspCW[ i ] << ( 10 - BitDepthY)) :
rspCW[ i ];
ChromaScaleCoef[ i ] = chromaResidualScaleLut[ Clip3(1, 64, binCW >> 1 ) - 1 ]
```

Proposed simplifications 3 and 4

```
shiftY = 11
ReshapePivot[ 0 ] = 0;
for( i = 0; i <= MaxBinIdx ; i++) {
    ReshapePivot[ i + 1 ] = ReshapePivot[ i ] + RspCW[ i ]
    ScaleCoeff[ i ] = ( RspCW[ i ] * (1 << shiftY) + (1 << (Log2(OrgCW) - 1))) >> (Log2(OrgCW))
    if ( RspCW[ i ] == 0 )
        InvScaleCoeff[ i ] = 0
    else
        InvScaleCoeff[ i ] = OrgCW * (1 << shiftY) / RspCW[ i ]
}
```

The variable ChromaScaleCoeff[ i ] with i in the range of 0 to MaxBinIdx , inclusive, are derived as follows:

```
shiftC = 11
- if ( RspCW[ i ] == 0 )

    ChromaScaleCoef [ i ] = (1 << shiftC)
- Otherwise (RspCW[ i ] != 0), the following applies:
ChromaScaleCoef[ i ] = InvScaleCoeff[ i ]
```

## Simplifications 3 and 4: Unification of calculations

### Simplification 3: Unification of fixed-point precision for luma and chroma scaling

#### Simplification 4: Unification of calculation of luma and chroma scaling factor

As adopted in VTM4

## Proposed simplifications 3 and 4

```

shiftY = 14
ReshapePivot[ 0 ] = 0;
for( i = 0; i <= MaxBinIdx ; i++) {
    ReshapePivot[ i + 1 ] = ReshapePivot[ i ] + rspCW[ i ]
    ScaleCoeff[ i ] = ( rspCW[ i ] * ( 1 << shiftY ) + ( 1 << (Log2(OrgCW) - 1))) >> (Log2(OrgCW))
    if ( RspCW[ i ] == 0 )
        InvScaleCoeff[ i ] = 0
    elseza
        InvScaleCoeff[ i ] = OrgCW * ( 1 << shiftY ) / rspCW[ i ]
}

```

The variable ChromaScaleCoef[ i ] with i in the range of 0 to MaxBinIdx, inclusive, are derived as follows:

```
chromaResidualScaleLut[64] = {16384, 16384, 16384, 16384, 16384, 16384, 16384, 8192, 8192, 8192, 8192, 8192, 8192, 5461, 5461, 5461, 5461, 4096, 4096, 4096, 4096, 4096, 3277, 3277, 3277, 3277, 2731, 2731, 2731, 2731, 2341, 2341, 2341, 2048, 2048, 2048, 1820, 1820, 1820, 1820, 1638, 1638, 1638, 1638, 1489, 1489, 1489, 1489, 1365, 1365, 1365, 1365, 1260, 1260, 1260, 1260, 1170, 1170, 1170, 1170, 1092, 1092, 1092, 1092, 1024, 1024, 1024, 1024};
```

**shiftC = 11**

- if ( RspCW[ i ] == 0 )
  - ChromaScaleCoef [ i ] = (1 << shiftC)
  - Otherwise (RspCW[ i ] != 0), the following applies:

$binCW = \text{BitDepth}_Y > 10 ? ( \text{rspCW}[ i ] \gg ( \text{BitDepth}_Y - 10 ) ) : \text{BitDepth}_Y < 10 ? ( \text{rspCW}[ i ] \ll ( 10 - \text{BitDepth}_Y ) ) : \text{rspCW}[ i ]$ ;

$\text{chromaScaleCoef}[ i ] = \text{chromaResidualScaleLut}[ \text{Clip3}(1, 64, binCW \gg 1) - 1 ]$

`shiftY = 11` ← Simplification 3: Unification of fixed-point precision

```

shiftY = 11
ReshapePivot[ 0 ] = 0;
for( i = 0; i <= MaxBinIdx ; i++) {
    ReshapePivot[ i + 1 ] = ReshapePivot[ i ] + RspCW[ i ]
    ScaleCoeff[ i ] = ( RspCW[ i ] * ( 1 << shiftY ) + ( 1 << ( Log2( OrgCW ) - 1 ) ) ) >> ( Log2( OrgCW ) )
    if ( RspCW[ i ] == 0 )
        InvScaleCoeff[ i ] = 0
    else
        InvScaleCoeff[ i ] = OrgCW * ( 1 << shiftY ) / RspCW[ i ]
}

```

The variable ChromaScaleCoef[ i ] with i in the range of 0 to MaxBinIdx , inclusive, are derived as follows:

### Simplification 4:

Unification of calculation of luma and chroma scaling factor

```

shiftC = 11
– if ( RspCW[ i ] == 0 )
    ChromaScaleCoef [ i ] = ( 1 << shiftC )
– Otherwise ( RspCW[ i ] != 0 ), the following applies:
    ChromaScaleCoef [ i ] = InvScaleCoef [ i ]

```

# Results of simplification 3 and 4

Simplification 3: Unification of fixed-point precision for luma and chroma scaling

Simplification 4: Unification of calculation of luma and chroma scaling factor

## Simplification 3 and 4 – Test results on SDR CTC over VTM4.0 anchors

	All Intra Main10			Random access Main10			Low delay B Main10			Low delay P Main10		
	Y	U	V	Y	U	V	Y	U	V	Y	U	V
Class A1	0.00%	0.00%	-0.01%	-0.02%	0.34%	0.22%						
Class A2	0.00%	0.00%	0.00%	0.04%	-0.38%	-0.41%						
Class B	0.00%	0.00%	0.00%	0.01%	-0.18%	0.02%	0.00%	-0.66%	-0.48%	0.02%	-0.68%	-0.50%
Class C	0.00%	0.00%	0.00%	0.00%	0.07%	0.04%	0.00%	-0.07%	-0.34%	0.04%	-0.52%	-0.64%
Class E	0.00%	0.01%	0.00%				-0.03%	1.28%	0.39%	-0.15%	1.04%	1.37%
<b>Overall</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.01%</b>	<b>-0.05%</b>	<b>-0.02%</b>	<b>-0.01%</b>	<b>0.02%</b>	<b>-0.22%</b>	<b>-0.02%</b>	<b>-0.20%</b>	<b>-0.08%</b>
Class D	0.00%	0.00%	0.00%	-0.04%	-0.27%	-0.19%	0.01%	-1.56%	-1.35%	0.06%	-0.75%	-0.87%
Class F	0.00%	0.00%	-0.01%	-0.04%	0.25%	0.17%	-0.05%	-0.04%	-0.04%	0.08%	-0.02%	0.03%

Change in overall BD-rate is very small

Enc/Dec time is similar as VTM4.0

# Results – Simplification 3 only

As adopted in VTM4:

Luma mapping and chroma residue scaling are implemented with fixed-point integer operations.

For luma mapping, the precision (shiftY) used to represent the fractional bits is 14.

For chroma residue scaling, the precision (shiftC) used to represent the fractional bits is 11.

Proposed simplification:

Unify the precision of luma mapping and chroma residue scaling:  $\text{shiftY} = \text{shiftC} = 11$

## Simplification 3 only – Test results on SDR CTC over VTM4.0 anchors

	All Intra Main10			Random access Main10			Low delay B Main10			Low delay P Main10		
	Y	U	V	Y	U	V	Y	U	V	Y	U	V
Class A1	0.00%	0.00%	-0.01%	0.00%	-0.03%	-0.03%						
Class A2	0.00%	0.00%	0.00%	-0.02%	-0.06%	0.00%						
Class B	0.00%	0.00%	0.00%	-0.01%	-0.04%	-0.04%	-0.01%	-0.20%	-0.26%	-0.02%	-0.18%	0.15%
Class C	0.00%	0.00%	0.00%	0.02%	-0.04%	-0.04%	-0.01%	0.14%	0.24%	0.00%	-0.43%	-0.35%
Class E	0.00%	0.01%	0.00%				-0.04%	-0.15%	-0.31%	0.05%	-0.24%	0.07%
<b>Overall</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>-0.04%</b>	<b>-0.03%</b>	<b>-0.02%</b>	<b>-0.07%</b>	<b>-0.10%</b>	<b>0.01%</b>	<b>-0.28%</b>	<b>-0.04%</b>
Class D	0.00%	0.00%	0.00%	-0.02%	-0.09%	-0.10%	-0.09%	0.65%	-0.48%	0.05%	-0.09%	0.17%
Class F	0.00%	0.00%	-0.01%	-0.02%	0.00%	-0.02%	-0.06%	-0.03%	0.24%	0.04%	-0.06%	0.08%

Change in overall BD-rate is very small

Enc/Dec time is similar as VTM4.0

# Simplification 4: Unification of luma/chroma scaling factor

- As adopted in VTM4:
  - Chroma residue scale is calculated using a pre-computed 64-entry codewords-to-scale mapping table:  

```
chromaResidualScaleLut[64] = {16384, 16384, 16384, 16384, 16384, 16384, 16384, 8192, 8192, 8192, 8192, 5461, 5461, 5461, 5461, 4096, 4096, 4096, 4096, 3277, 3277, 3277, 3277, 2731, 2731, 2731, 2731, 2341, 2341, 2341, 2048, 2048, 2048, 1820, 1820, 1820, 1638, 1638, 1638, 1638, 1489, 1489, 1489, 1489, 1365, 1365, 1365, 1260, 1260, 1260, 1260, 1170, 1170, 1170, 1170, 1092, 1092, 1092, 1092, 1024, 1024, 1024, 1024};
```
  - The chroma scale  $\text{ChromaScaleCoef}[i]$  for  $i$ -th luma PWL piece is calculated as:  
$$\text{binCW} = \text{BitDepthY} > 10 ? (\text{rspCW}[i] \gg (\text{BitDepthY} - 10)) : \text{BitDepthY} < 10 ? (\text{rspCW}[i] \ll (10 - \text{BitDepthY})) : \text{rspCW}[i]$$
$$\text{ChromaScaleCoef}[i] = \text{chromaResidualScaleLut}[\text{Clip3}(1, 64, \text{binCW} \gg 1) - 1]$$
- Proposed simplification:
  - Remove use of  $\text{chromaResidualScaleLut}[]$  and calculate the chroma residue scale in the same way that inverse luma scale is calculated ( $\text{InvScaleCoeff}[i] = \text{OrgCW} * (1 \ll \text{shiftY}) / \text{binCW}[i]$ ):  
$$\text{ChromaScaleCoef}[i] = \text{OrgCW} * (1 \ll \text{shiftC}) / \text{binCW}[i]$$
  - When both simplification 3 ( $\text{shiftY} = \text{shiftC} = 11$ ) and 4 are enabled, for valid pieces  
$$\text{ChromaScaleCoef}[i] = \text{InvScaleCoeff}[i]$$

# Results of simplification 4 only

Simplification 4: Unification of calculation of luma and chroma scaling factor

## Simplification 4 only – Test results on SDR CTC over VTM4.0 anchors

	All Intra Main10			Random access Main10			Low delay B Main10			Low delay P Main10		
	Y	U	V	Y	U	V	Y	U	V	Y	U	V
Class A1	0.00%	0.00%	0.00%	-0.01%	0.34%	0.24%						
Class A2	0.00%	0.00%	0.00%	0.03%	-0.49%	-0.46%						
Class B	0.00%	0.00%	0.00%	0.01%	-0.15%	0.01%	0.01%	-0.68%	-0.55%	-0.01%	-0.72%	-0.40%
Class C	0.00%	0.00%	0.00%	-0.02%	0.08%	-0.03%	-0.02%	-0.34%	-0.42%	0.04%	-0.53%	-0.42%
Class E	0.00%	0.00%	0.00%				-0.03%	0.96%	0.49%	-0.16%	0.95%	1.20%
<b>Overall</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>-0.06%</b>	<b>-0.05%</b>	<b>-0.01%</b>	<b>-0.16%</b>	<b>-0.24%</b>	<b>-0.03%</b>	<b>-0.24%</b>	<b>-0.01%</b>
Class D	0.00%	0.00%	0.00%	-0.02%	-0.31%	-0.29%	-0.01%	-0.43%	-0.80%	0.08%	-0.53%	-0.85%
Class F	0.00%	0.00%	0.00%	-0.03%	0.21%	0.19%	0.04%	-0.06%	0.11%	0.01%	0.10%	0.03%

Change in overall BD-rate is very small

Enc/Dec time is similar as VTM4.0

# Results of all implementation simplifications: 1, 2, 3, and 4

**Simplification 1, 2, 3 and 4 – Test results on SDR CTC over VTM4.0 anchors**

	All Intra Main10			Random access Main10			Low delay B Main10			Low delay P Main10		
	Y	U	V	Y	U	V	Y	U	V	Y	U	V
Class A1	0.00%	0.00%	-0.01%	0.00%	0.35%	0.36%						
Class A2	0.00%	0.00%	0.00%	0.03%	-0.38%	-0.44%						
Class B	0.00%	0.00%	0.00%	0.01%	-0.17%	-0.06%	0.04%	-0.64%	-0.92%	0.02%	-0.72%	-0.21%
Class C	0.00%	0.00%	0.00%	-0.02%	0.03%	0.12%	-0.02%	-0.20%	-0.40%	0.03%	-0.56%	-0.64%
Class E	0.00%	0.01%	0.00%				0.02%	0.96%	0.36%	-0.10%	1.17%	0.92%
<b>Overall</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.00%</b>	<b>0.01%</b>	<b>-0.05%</b>	<b>-0.01%</b>	<b>0.01%</b>	<b>-0.09%</b>	<b>-0.42%</b>	<b>-0.01%</b>	<b>-0.19%</b>	<b>-0.07%</b>
Class D	0.00%	0.00%	0.00%	-0.01%	-0.38%	-0.13%	-0.05%	-0.93%	-0.94%	0.03%	-1.17%	-0.65%
Class F	0.00%	0.00%	-0.01%	-0.04%	0.22%	0.20%	-0.03%	-0.20%	-0.05%	-0.01%	0.09%	-0.04%

**Simplification 1, 2, 3 and 4 – Test results on HDR over VTM4.0 HDR reshaping  
(LumaDQP=0, LMCS=1, SignalType=HDR-PQ)**

	All Intra Main10			Random Access Main10		
	wPsnrY	DE100	PSNRL100	wPsnrY	DE100	PSNRL100
<b>Average HDR-B</b>	<b>0.00%</b>	<b>0.01%</b>	<b>0.01%</b>	<b>0.06%</b>	<b>-0.18%</b>	<b>0.06%</b>

Change in overall BD-rate is very small.  
Enc/Dec time is similar as VTM4.0.



# Overview

- Introduction
- Simplification of LMCS implementation (normative)
  - Reduced pipeline delay
  - Reduced local buffer
  - Unified luma and chroma fixed-point precision
  - Unified luma and chroma scaling factor calculation method
- Non-CTC non-normative topics (informative)
  - VTM modification for other bit depths
  - Interaction of LMCS and local QP adjustment
  - SCC TGM results
- Conclusion

# LMCS for various InternalBitDepth

- CTC: InternalBitDepth = 10
- Software improved to support other bitdepths
  - Decoder support is straightforward
  - Encoder: some simple adjustment
    - SDR: re-uses 10b codeword allocation scheme by re-normalizing picture statistics to 10b, then shifts codewords for each piece to targeted InternalBitDepth
    - HDR:
      - HDR anchor: fix lumaDQP table by shifting luma value range from 10b to the targeted bitdepth and reusing same dQP value corresponding to 10b. (merge request #364)
      - LMCS: change fixed-point 10b implementation to floating point implementation for any bitdepth (merge request #365)
        - HDR 10b results same as fixed-point implementation.
- Simulation results:
  - SDR: Show results for 8b (8bit CTC clips), 12b (all CTC clips)
  - HDR: show results for 12b

# Simulation results (RA): BitDepth Test

	SDR: RA-8		
	LMCS=1 Over LMCS=0		
	Y	U	V
Class A1			
Class A2			
Class B			
Class C	-0.60%	-0.04%	0.03%
<b>Overall</b>			
Class D	-0.50%	-1.81%	-1.25%
Class F	-0.79%	0.65%	0.78%

	SDR: RA-10		
	LMCS=1 Over LMCS=0		
	Y	U	V
Class A1	-0.96%	3.12%	2.91%
Class A2	-1.16%	2.77%	2.90%
Class B	-1.61%	1.03%	0.77%
Class C	-1.44%	3.86%	4.01%
<b>Overall</b>	<b>-1.34%</b>	<b>2.55%</b>	<b>2.48%</b>
Class D	-1.20%	3.98%	4.18%
Class F	-1.67%	2.53%	2.45%

	SDR: RA-12		
	LMCS=1 Over LMCS=0		
	Y	U	V
Class A1	-1.08%	3.69%	3.50%
Class A2	-1.37%	3.30%	3.40%
Class B	-1.70%	1.59%	1.28%
Class C	-1.51%	4.51%	4.41%
<b>Overall</b>	<b>-1.46%</b>	<b>3.13%</b>	<b>2.98%</b>
Class D	-1.45%	4.02%	4.51%
Class F	-1.78%	2.58%	2.53%

**Note: class A1/A2/B has native 10bit input video and are only tested with bitdepth >=10 coding**

	HDR: RA-10		
	DE100	L100	wPsnrY
<b>HDR-B</b>	<b>7.7%</b>	<b>-1.9%</b>	<b>-1.8%</b>

	HDR: RA-12		
	DE100	L100	wPsnrY
<b>HDR-B</b>	<b>8.6%</b>	<b>-2.1%</b>	<b>-1.9%</b>

# LMCS and local QP adjustment for SDR

- Coding gains from LMCS and from local QP adjustment are independent and additive
  - LMCS: fine-granularity pixel-level process
  - Local QP adjustment: a coarser block-level transform-domain process
- Simulation:
  - Measure coding gains relative to VTM4.0 of the LMCS and of rate-distortion optimized local QP selection (e.g., testing 0/+/-1 values of QP) independently and in combination
  - Local QP adjustment: MaxDeltaQP=1
    - Encoding time: 2.5x of VTM4.0 anchor => Class B/C/D/F for RA case
  - Test cases:
    - “MaxDeltaQP=1 and LMCS=0” versus “MaxDeltaQP=0 and LMCS=0”;
    - “MaxDeltaQP=0 and LMCS=1” versus “MaxDeltaQP=0 and LMCS=0”;
    - “MaxDeltaQP=1 and LMCS=1” versus “MaxDeltaQP=0 and LMCS=0”.

# BDRate of LMCS and local QP adjustment individually and combined under RA (Class B/C/D/F)

	RA-10			
	"MaxDeltaQP=1, LMCS=0" vs "MaxDeltaQP=0, LMCS=0"			
	Y	U	V	EncT
Class B	-0.52%	-1.22%	-1.26%	249%
Class C	-0.39%	-0.89%	-1.15%	251%
Class D	-0.33%	-0.71%	-0.58%	265%
Class F	-0.68%	-0.80%	-0.85%	244%
<b>Average</b>	<b>-0.48%</b>	<b>-0.91%</b>	<b>-0.96%</b>	<b>252%</b>

	RA-10			
	"MaxDeltaQP=0, LMCS=1" vs "MaxDeltaQP=0, LMCS=0"			
	Y	U	V	EncT
Class B	-1.61%	1.03%	0.77%	107%
Class C	-1.44%	3.86%	4.01%	104%
Class D	-1.20%	3.98%	4.18%	110%
Class F	-1.67%	2.53%	2.45%	103%
<b>Average</b>	<b>-1.48%</b>	<b>2.85%</b>	<b>2.85%</b>	<b>106%</b>

	RA-10			
	"MaxDeltaQP=1, LMCS=1" vs "MaxDeltaQP=0, LMCS=0"			
	Y	U	V	EncT
Class B	-2.11%	-0.40%	-0.60%	269%
Class C	-1.80%	2.74%	2.89%	252%
Class D	-1.50%	3.01%	3.39%	287%
Class F	-2.36%	1.70%	1.71%	256%
<b>Average</b>	<b>-1.94%</b>	<b>1.76%</b>	<b>1.85%</b>	<b>266%</b>

**Difference of combined LMCS and local QP adjustment and sum of LMCS and local QP adjustment individually under RA (Class B/C/D/F)**

	RA10		
	Difference		
	Y	U	V
Class B	0.02%	-0.21%	-0.11%
Class C	0.03%	-0.23%	0.03%
Class D	0.03%	-0.26%	-0.21%
Class F	-0.01%	-0.03%	0.11%
<b>Average</b>	<b>0.02%</b>	<b>-0.18%</b>	<b>-0.04%</b>

# LMCS and local QP adjustment for HDR

- Two settings in VTM4.0 for local QP adjustment
  - maxDeltaQP: RDO optimization for various dQP value
  - lumaDQP: designed for HDR based on luma value,
    - Weight in wPSNR metrics matches lumaDQP:  $\text{weight} = 2^{(dQP/3)}$
  - In VTM4.0, maxDeltaQP and lumaDQP cannot be enabled at same time
- Simulations:
  - “MaxDeltaQP=1 and LMCS=1” versus “MaxDeltaQP=0 and LMCS=1”
  - Local QP adjustment provides additional gain for HDR.

RA-10	DE100	PSNR	L100	wPsnrY	wPsnrU	wPsnrV	psnrY	psnrU	psnrV
HDR-B	-1.54%	-0.49%	-0.26%	-1.91%	-2.63%	-0.19%	-1.86%	-2.50%	

# LMCS and lumaDQP adjustment for HDR

- LMCS and lumaDQP can be used together
  - JVET-J0015: subjective benefit
- LMCS default curve and lumaDQP default table are not intended to be used together (no support in VTM)
  - Weight in wPSNR metrics is derived using lumaDQP table:  $\text{weight}_{\text{dQP}} = 2^{(\text{dQP}/3)}$
  - LMCS curve is designed to match weight in wPSNR metrics:  $\text{slope} = \sqrt{\text{weight}_{\text{dQP}}}$
  - LMCS + lumaDQP results in  $\text{weight}_{\text{dQP}} * \text{weight}_{\text{dQP}} \Rightarrow$  not optimized for wPSNR metrics.
- Simulation (Quick “what-if” hack in VTM4.0 code):

	AI (lumaDQP+LMCS compared to LMCS)							
	DE100	PSNRL100	wPsnrY	wPsnrU	wPsnrV	psnrY	psnrU	psnrV
Average HDR-B	4.98%	2.12%	4.92%	4.25%	15.58%	8.53%	8.31%	24.44%



# SCC TGM LMCS Results

- SCC TGM content:
  - used in CE8 SCC test: 1920x1080 8b content
  - FlyingGraphics, Desktop, Console, ChineseEditing

SCC TGM	VTM4.0 (LMCS=1) over LMCS=0		
	Y	U	V
AI 10	-1.36%	0.00%	-0.23%
RA 10	-1.44%	0.15%	0.09%
LDB 10	-1.67%	-0.10%	-0.19%
LDP 10	-2.12%	-0.40%	-0.41%

# Conclusion

- Proposes four LMCS simplifications (for #1, see also N0806)
  1. reduction of the pipeline delay for computing average luma for chroma residue scaling => use upper-left luma value
  2. reduction of the size of the local buffer needed to store chroma residue samples => add clipping on residue signal before scaling
  3. unification of the fixed-point precision used in luma mapping and chroma residue scaling=>  $\text{shiftY} = \text{shiftC} = 11$
  4. unification of the method of calculating chroma residue scale with the method of calculating the luma inverse scale, and removal of the pre-computed LUT
    - Performance of simplifications are individually and collectively very close to the VTM4.0 anchor.
    - Suggested for adoption.
- Additional informative non-CTC results are described.
  - Continue to improve encoder algorithm.

# Acknowledgement

- We would like to thank Technicolor for crosschecking the test results (JVET-N0526).

# ClipRange

- Text currently Versatile Video Coding (Draft 4) (JVET-M1001-v7): LMCS

The variables ClipRange, LmcsMinVal, and LmcsMaxVal are derived as follows:

$$\text{ClipRange} = ((\text{lmc\_min\_bin\_idx} > 0) \ \&\& \ (\text{LmcsMaxBinIdx} < 15)) \quad (7-77)$$

$$\text{LmcsMinVal} = 16 \ll (\text{BitDepth}_Y - 8) \quad (7-78)$$

$$\text{LmcsMaxVal} = 235 \ll (\text{BitDepth}_Y - 8) \quad (7-79)$$

- Reason for implicit range clipping:
  - Assumption: if 0<sup>th</sup> piece and 15<sup>th</sup> piece have no codewords, it is standard range.



[0 1023] in 16 equal pieces

- If standard range is assumed, clip to a value of 940 after inverse reshaping

# ClipRange

- Test: Remove implicit clipping

~~The variables ClipRange, LmcsMinVal, and LmcsMaxVal are derived as follows:~~

~~$$\text{ClipRange} = ((\text{lmc\_min\_bin\_idx} > 0) \&\& (\text{LmcsMaxBinIdx} < 15)) \quad (7\_77)$$~~

~~$$\text{LmcsMinVal} = 16 \ll (\text{BitDepth}_Y = 8) \quad (7\_78)$$~~

~~$$\text{LmcsMaxVal} = 235 \ll (\text{BitDepth}_Y = 8) \quad (7\_79)$$~~

- Test results:

	All Intra Main10		
	Over VTM 4.0		
	Y	U	V
Class A1	0.03%	0.01%	-0.01%
Class A2	0.00%	0.00%	0.00%
Class B	0.06%	0.00%	0.00%
Class C	0.04%	0.00%	0.01%
Class E	0.02%	0.00%	0.00%
<b>Overall</b>	0.03%	0.00%	0.00%
Class D	0.12%	0.00%	0.01%
Class F	0.09%	0.00%	0.00%

	Random access Main10		
	Over VTM 4.0		
	Y	U	V
Class A1	0.03%	0.00%	-0.01%
Class A2	-0.01%	-0.12%	0.01%
Class B	0.15%	-0.03%	-0.13%
Class C	0.07%	-0.04%	-0.08%
Class E			
<b>Overall</b>	0.07%	-0.04%	-0.06%
Class D	0.25%	-0.06%	-0.08%
Class F	0.07%	0.00%	-0.02%



# JVET-N0220: AHG16: Simplification of LMCS (Reshaper) Implementation

---

Mar. 2019

Dolby Laboratories