JCTVC-O0228

MEDIATEK

AHG9: Signaling lossless slices

Shih-Ta Hsiang, Yu-Wen Huang, and Shawmin Lei

Presented by Yu-Wen Huang
15th JCT-VC Meeting in Geneva
23 Oct. – 1 Nov. 2013

Overall Summary

- **Objective**
  Provide a simple mechanism to clearly and compactly indicate an entire slice in a lossless representation

- **Proposed Method**
  - Add a new syntax flag in the slice segment header to indicate the use of the lossless coding mode for the entire slice
  - Add a PPS flag to enable/disable this slice header flag

- **Features**
  - Compactly signal lossless coding for an entire slice by one flag
  - Exclude tools and resources dedicated to lossy coding only
    - Context reduction for significant_coeff_flag from 44 to 2
  - Explicitly remove the syntax information irrelevant to lossless coding
  - No modifications to the current HEVC lossy bitstreams

Introduction

- Lossless signaling in the current HEVC RExt Draft
  - PPS: **transquant_bypass_enabled_flag**
  - CU: **cu_transquant_bypass_flag**

- Selection of lossless coding in the current HM & HM-RExt reference software
  - Encoder option **transquantBypassEnableFlag** = 1
    - HM encoder sets PPS-> transquant_bypass_enabled_flag equal to 1
  - Encoder option **CUTransquantBypassFlagForce** = 1
    - HM encoder exclusively sets each CU in the lossless mode and codes cu_transquant_bypass_flag equal to 1 for each CU
    - No single syntax flag in the spec corresponding to this option

Proposed Modifications

- PPS extension
  - Code new syntax **cu_transquant_bypass_forced_present_flag** when transquant_bypass_enabled_flag is equal to 1
- Slice segment header
  - Code new syntax **cu_transquant_bypass_forced_flag** when cu_transquant_bypass_forced_present_flag is equal to 1
  - When cu_transquant_bypass_forced_flag is equal to 1, the syntax elements related to SAO, quantization, and deblocking are bypassed in the current slice
- Coding unit
  - **cu_transquant_bypass_flag** is not coded and inferred to be 1 when both transquant_bypass_enabled_flag & cu_transquant_bypass_forced_flag are equal to 1

Proposed Picture Parameter Set

pic_parameter_set_rbsp( ) {	**Descriptor**
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
. . . . .	
slice_segment_header_extension_present_flag	u(1)
pps_extension1_flag	u(1)
if(pps_extension1_flag) {	
if(transform_skip_enabled_flag)	
log2_transform_skip_max_size_minus2	ue(v)
if(transquant_bypass_enabled_flag)	
cu_transquant_bypass_forced_present_flag	u(1)
pps_extension2_flag	u(1)
}	
if(pps_extension2_flag)	
while(more_rbsp_data( ))	
pps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

Proposed Slice Header Syntax

slice_segment_header( ) {	**Descriptor**
first_slice_segment_in_pic_flag	u(1)
. . . .	
if(!dependent_slice_segment_flag) {	
. . . .	
if(cu_transquant_bypass_forced_present_flag)	
cu_transquant_bypass_forced_flag	u(1)
if(sample_adaptive_offset_enabled_flag && !cu_transquant_bypass_forced_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
. . . .	
if(!cu_transquant_bypass_forced_flag || pps_loop_filter_across_slices_enabled_flag)	
slice_qp_delta	se(v)
if(!cu_transquant_bypass_forced_flag) {	
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
}	
if(deblocking_filter_override_enabled_flag)	
deblocking_filter_override_flag	u(1)
if(deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
}	
}	
if(pps_loop_filter_across_slices_enabled_flag && (slice_sao_luma_flag || slice_sao_chroma_flag || !slice_deblocking_filter_disabled_flag))	
slice_loop_filter_across_slices_enabled_flag	u(1)
}	
. . . .	
}	

MEDIATEK

Proposed Coding Unit & Transform Unit Syntax

coding_unit(x0, y0, log2CbSize) {	**Descriptor**
if(transquant_bypass_enabled_flag && !cu_transquant_bypass_forced_flag)	
cu_transquant_bypass_flag	ae(v)
. . . .	

transform_unit(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	**Descriptor**
log2TrafoSizeC = log2TrafoSize − (ChromaArrayType = = 3 ? 0 : 1)	
if(cbf_luma[x0][y0][trafoDepth] || cbf_cb[x0][y0][trafoDepth] || cbf_cr[x0][y0][trafoDepth] || (ChromaArrayType = = 2 && (cbf_cb[x0][y0 + (1 << log2TrafoSizeC)][trafoDepth] || cbf_cr[x0][y0 + (1 << log2TrafoSizeC)][trafoDepth]))) {	
if(cu_qp_delta_enabled_flag && !IsCuQpDeltaCoded && !cu_transquant_bypass_forced_flag) {	
cu_qp_delta_abs	ae(v)
if(cu_qp_delta_abs)	
cu_qp_delta_sign_flag	ae(v)
}	
. . . .	
}	
}	

Conclusion & Recommendation

- Proposed the high-level syntax modifications to support signaling a lossless representation for an entire slice
- Benefits
  - More compact in signaling and more concise in concept for common lossless video coding applications
  - Can save the resources dedicated to lossy coding when coding a lossless slice
  - Explicitly exclude the irrelevant syntax information
    - Bypass coding syntax elements related to SAO, quantization, and deblocking in the slice header
    - Ensure CTU SAO parameters & CU delta QP parameters will never be included in the bitstream (CTU SAO parameters are coded under the current lossless SCC CTCs)
- Recommend adoption into the next HEVC RExt draft