



JCTVC-E279: Extensible High Layer Syntax for Scalability

Jill Boyce, Danny Hong, Stephan Wenger

Vidyo



Background and Introduction

- **When H.264/AVC was originally designed, scalability and multiview coding were not included**
 - Backwards-compatible retrofit needed
- **Considering scalability now allows for cleaner high level design**
- **Several high level syntax changes proposed**
 - Improve bitstream extraction operations in gateways
 - Provide additional information to decoders
 - Improve parallel decoding
 - Decrease delays by allowing decoding of picture to start earlier

Proposed Changes Summary

- 1. Dependency Parameter Set**
- 2. NAL Unit header**
- 3. Sequence Parameter Set**
- 4. Picture Parameter Set**
- 5. Temporal structure SEI message**
- 6. Spatial quality layer SEI message**
- 7. Slice Header**

1. Dependency Parameter Set

- **Introduce new parameter set, above Sequence Parameter Set, to associate multiple related sequences**
- **Can be used for scalability, multi-view, or future extensions (e.g. multiple description coding)**
- **Scalability parameters proposed**
 - Define number of spatial and quality layers
 - Define spatial dependency relationship
 - Limit dependency to immediately lower spatial layer, and to top quality layer
- **Parameters for other extensions undefined**
- **Helpful for bitstream extractors**
 - Required, unlike Scalability info SEI message
 - May be sent in advance, out-of-band
- **Open to other naming for this parameter set**

1. Dependency Parameter Set

dep_parameter_set_rbsp() {	Descriptor
dep_parameter_set_id	u(2)
scalability_flag	u(1)
if(scalability_flag) {	
max_spatial_layers_minus1	u(3)
for(i = 0; i <= max_spatial_layers_minus1; i++) {	
dependency_flag[i]	u(1)
max_quality_layers_minus_1[i]	u(4)
}	
}	
multiview_flag	u(1)
if(multiview_flag) {	
// undefined	
}	
dps_vui_parameters_present_flag	u(1)
if(dps_vui_parameters_present_flag) {	
// undefined	
}	
extension_flag	u(1)
if(extension_flag) {	
// undefined	
}	
}	

dep_parameter_set_id identifies the dependency parameter set that is referred to by the sequence parameter set. The value of dep_parameter_set_id shall be in the range of 0 to 3, inclusive.

scalability_flag equal to 1 specifies that multiple spatial and/or quality layers may be present. scalability_flag equal to 0 specifies that only a single spatial layer and single quality layer is present.

max_spatial_layers_minus1 + 1 specifies the maximum number of spatial layers that may be present. max_spatial_layers_minus1 shall be in the range of 0 to 7, inclusive.

dependency_flag[i] specifies whether inter-layer prediction from the highest quality layer of the coded sequence with spatial_id equal to i may be used for decoding the coded slice with spatial_id equal to i + 1. If dependency_flag[i] is equal to 1, inter-layer prediction may be used. If dependency_flag[i] is equal to 0, inter-layer prediction may not be used.

max_quality_layers_minus_1 + 1 specifies the maximum number of quality layers that may be present for the spatial layer with spatial_id equal to i. max_quality_layers_minus_1 shall be in the range of 0 to 15, inclusive.

multiview_flag shall be equal to 0.

extension_flag shall be equal to 0.

2. NAL Unit Header

- Extensions similar to SVC, dependent on `nal_unit_type` values
 - Add fields for `spatial_id` and `quality_id`, similar to in SVC
 - Add fields for `store_ref_base_pic_flag` and `use_ref_base_pic_flag`

<code>nal_unit(NumBytesInNALunit) {</code>	Descriptor
<code>forbidden_zero_bit</code>	<code>f(1)</code>
<code>nal_ref_idc</code>	<code>u(2)</code>
<code>nal_unit_type</code>	<code>u(5)</code>
<code>NumBytesInRBSP = 0</code>	
<code>nalUnitHeaderBytes = 1</code>	
<code>if(nal_unit_type == 1 nal_unit_type == 5) {</code>	
<code>temporal_id</code>	<code>u(3)</code>
<code>output_flag</code>	<code>u(1)</code>
<code>store_ref_base_pic_flag</code>	<code>u(1)</code>
<code>use_ref_base_pic_flag</code>	<code>u(1)</code>
<code>reserved_zero_2bits</code>	<code>u(2)</code>
<code>nalUnitHeaderBytes += 1</code>	
<code>}</code>	
<code>if(nal_unit_type == ? nal_unit_type == ?) {</code>	
<code>spatial_id</code>	<code>u(3)</code>
<code>quality_id</code>	<code>u(4)</code>
<code>reserved_zero_bit</code>	<code>u(1)</code>
<code>nalUnitHeaderBytes += 1</code>	
<code>}</code>	
<code>...</code>	

3. Sequence Parameter Set

- **Add syntax elements for dependency parameter set id and max number of temporal layers**
- **Add temporal_id_nesting_flag (as proposed in JCTVC-D200)**
 - Mark as “unused for reference” all pictures in the reference picture list with higher values of temporal_id
 - Software provided for this feature
 - Software also includes fixes for some other high level syntax inconsistencies between WD and HM2.0, related to temporal_id
- **Add SPS extension flag**
 - For scalability parameters, or future extensions

3. Sequence Parameter Set

seq_parameter_set_rbsp() {	Descriptor
seq_parameter_set_id	u(5)
dep_parameter_set_id	u(2)
reserved_zero_bit	u(1)
profile_idc	u(8)
level_idc	u(8)
max_temporal_layers_minus1	u(3)
pic_width_in_luma_samples	u(16)

interpolation_filter_flag	u(1)
temporal_id_nesting_flag	u(1)
sps_extension_flag	u(1)
if(sps_extension_flag) {	u(1)
//undefined	
}	
rbp_trailing_bits()	
}	

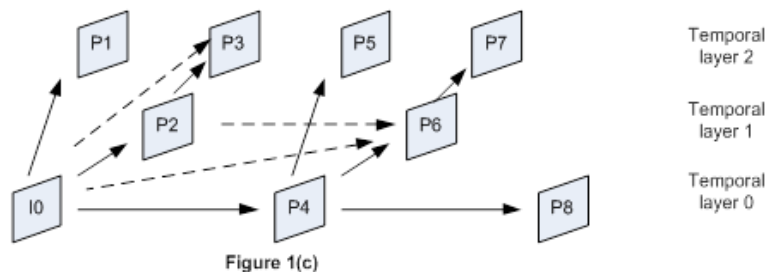
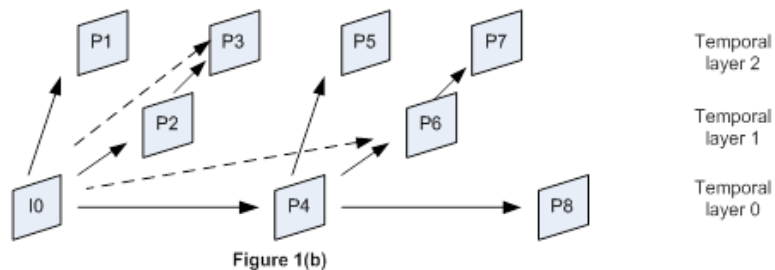
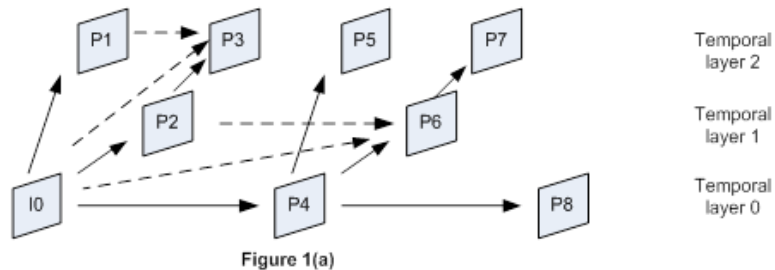
4. Picture Parameter Set

- **Add temporal_layer_switching_point_flag**
 - Similar to proposal in JCTVC-D200, but in PPS not NAL unit header
 - Instead, add separate flag for each temporal level
 - If temporal_switching_point_flag[i] is equal to 1, all pictures with values of temporal_id higher than i in the reference picture storage are marked as “unused for reference”
 - Software provided for this feature
- **Add pps_extension_flag**
 - For scalability parameters, or future extensions

4. Picture Parameter Set

pic_parameter_set_rbsp() {	Descriptor
pic_parameter_set_id	u(5)
seq_parameter_set_id	u(5)
entropy_coding_mode_flag	u(1)
temporal_layer_switching_params_present_flag	u(1)
if (temporal_layer_switching_params_present_flag == 1) {	
for(i = 0; i < max_temporal_layers_minus1; i++)	
temporal_layer_switching_point_flag[i]	u(1)
}	
num_ref_idx_l0_default_active_minus1	ue(v)
num_ref_idx_l1_default_active_minus1	ue(v)
pic_init_qp_minus26 /* relative to 26 */	se(v)
constrained_intra_pred_flag	u(1)
rbbsp_trailing_bits()	
pps_extension_flag	u(1)
if(pps_extension_flag) {	
//undefined	
}	
}	

Example: Improved flexibility with per layer temporal_switching_point_flag



For pictures P3 and P6

- Solid lines first predictor, dashed lines additional multiple ref pictures
- 1(a): temporal_id_nesting_flag is 0
 - P3 predictors: P2, P1, I0
 - P6 predictors: P4, P2, I0
- 1(b) temporal_id_nesting_flag is 1
 - P3 predictors: P2, ~~P1~~, I0
 - P6 predictors: P4, ~~P2~~, I0
- 1(c): temporal_id_nesting_flag is 0, P2's temporal_switching_point_flag[1] is 1, P4's temporal_switching_point_flag[0] is 0
 - P3 predictors: P2, ~~P1~~, I0
 - P6 predictors: P4, P2, I0

Create switching point for layer 1->2 switching, but not layer 0->1 switching

5. Temporal Structure SEI Message

- Add active # of temporal layers to the coding structure SEI message proposed in JCTVC-D200
- Useful for concatenated gateways and decoders to know how many temporal layers to expect
- Fixed width for easy updating by gateway
- Optional coding structure information in same message

temporal_structure(payloadSize) {	Descriptor
num_active_temporal_layers_minus1	u(3)
coding_structure_flag	u(1)
if(coding_structure_flag) {	
num_pictures_in_sop_minus1	ue(v)
num_sops_in_gop	ue(v)
for(i = 0; i < num_pictures_in_sop_minus1 + 1; i++) {	
primary_pic_type[i]	u(2)
ref_flag[i]	u(1)
temporal_num[i]	u(3)
display_num[i]	ue(v)
}	
}	
coding_statistics_flag	u(1)
if(coding_statistics_flag) {	
average_frame_rate_flag	u(1)
average_bit_rate_flag	u(1)
if(average_frame_rate_flag)	
average_frame_rate	u(16)
if(average_bit_rate_flag)	
for(i = 0; i < NumTemporalLayers; i++)	
average_bit_rate[i]	u(16)
}	
}	

6. Spatial quality layer SEI message

- **Indicate active # of spatial layers in a bitstream, and active # of quality layers per spatial layer**
 - Useful for concatenated gateways and decoders to know how many layers to expect
 - Fixed width for easy updating by gateway

<code>spatial_quality_layers(payloadSize) {</code>	Descriptor
<code> num_active_spatial_layers_minus1</code>	<code>u(3)</code>
<code> num_active_quality_layers_minus1</code>	<code>u(4)</code>
<code>}</code>	

num_active_spatial_layers_minus1 + 1 specifies how many spatial layers are present in the bitstream. num_active_spatial_layers_minus1 shall be in the range of 0 to max_spatial_layers_minus1, inclusive.

num_active_quality_layers_minus1 + 1 specifies how many quality layers are present in the spatial layer with spatial_id equal to num_active_spatial_layers_minus1. num_active_quality_layers_minus1 shall be in the range of 0 to max_quality_layers_minus1 [num_active_spatial_layers_minus1], inclusive.

7. Slice Header

- **Move pic_parameter_set_id field to the first element in the slice header**
 - More easily accessed by a gateway, without having to parse additional syntax elements
 - Has more impact on gateway operations than slice_type or first_tb_in_slice
- **Add slice_header_extension_flag**
 - For scalability parameters, or future extensions

slice_header() {	Descriptor
pic_parameter_set_id	ue(v)
first_tb_in_slice	ue(v)
slice_type	ue(v)
...	
slice_header_extension_flag	u(1)
if(slice_header_extension_flag) {	
//undefined	
}	
}	

Applicability

• **Proposals relevant to temporal scalability**

- 3. Sequence parameter set
- 4. Picture parameter set
- 5. Temporal structure SEI message
- 7. Slice header

• **Proposals relevant to spatial & quality scalability**

- 1. Dependency parameter set
- 2. NAL unit header
- 3. Sequence parameter set
- 4. Picture parameter set
- 6. Spatial quality layer SEI message
- 7. Slice header