AV1 Layer-specific OBUs

Wan-Teh Chang <wtc@google.com>
2024-04-08

Two pull requests

By eleft (Alex Eleftheriadis), merged on Sep 9, 2019

1. RTC SG clarifications on high-level syntax:

https://github.com/AOMediaCodec/av1-spec/pull/281

Commit 86fb0ac

2. OBU ordering clarification

https://github.com/AOMediaCodec/av1-spec/pull/280

Commit 27521e1

OBU header

```
obu_header() {                              Type
    ..
    obu_extension_flag                      f(1)
    ..
    if ( obu_extension_flag == 1 )
        obu_extension_header()
}

obu_extension_header() {                    Type
    temporal_id                             f(3)
    spatial_id                              f(2)
    extension_header_reserved_3bits         f(3)
}
```

Layers

Layer
A set of tile group OBUs with identical spatial_id and identical temporal_id values.

Base layer
The layer with **spatial_id** and **temporal_id** values equal to 0.

Enhancement layer
A layer with either spatial_id greater than 0 or temporal_id greater than 0.

The drop_obu() condition

```
if ( obu_type != OBU_SEQUENCE_HEADER &&
     obu_type != OBU_TEMPORAL_DELIMITER &&
     OperatingPointIdc != 0 &&
     obu_extension_flag == 1 )
{
    inTemporalLayer = (OperatingPointIdc >> temporal_id ) & 1
    inSpatialLayer = (OperatingPointIdc >> ( spatial_id + 8 ) ) & 1
    if ( !inTemporalLayer || ! inSpatialLayer ) {
        drop_obu( )
        Return
    }
}
```

6.2.2. OBU header semantics

The **obu_type** table gets a new “Layer-specific” column.

Three new paragraphs after the **obu_type** table:

- The column “Layer-specific” indicates if the corresponding OBU type is considered to be associated with a specific layer (“Y”), or not (“N”). ***OBUs that are not layer-specific must have the obu_extension_flag set to 0.***
- Metadata OBU types may or may not be layer-specific, depending on the metadata type. The table in Section 6.7.1 specifies which types of metadata OBUs are layer-specific and which are not.
- Padding OBUs may or may not be layer-specific. If the obu_extension_flag is set to 1 they are layer-specific, whereas when the obu_extension_flag is set to 0 they are not.

6.2.3. OBU extension header semantics

temporal_id specifies the temporal level of the data contained in the OBU. ***In layer-specific OBUs,*** when temporal_id is not present it is inferred to be equal to 0.

spatial_id specifies the spatial level of the data contained in the OBU. ***In layer-specific OBUs,*** when spatial_id is not present it is inferred to be equal to 0.

6.4.1. General sequence header OBU semantics

If operating_point_idc[op] is not equal to 0 for any value of op from 0 to operating_points_cnt_minus_1, it is a requirement of bitstream conformance that obu_extension_flag is equal to 1 ***for all layer-specific OBUs in the coded video sequence***.

Unchanged:

It is a requirement of bitstream conformance that if OperatingPointIdc is equal to 0, then obu_extension_flag is equal to 0 for all OBUs that follow this sequence header until the next sequence header.

7.5. Ordering of OBUs

Unchanged:

If a coded video sequence contains at least one enhancement layer (OBUs with spatial_id greater than 0 or temporal_id greater than 0) then all frame headers and tile group OBUs associated with base (spatial_id equals 0 and temporal_id equals 0) and enhancement layer (spatial_id greater than 0 or temporal_id greater than 0) data must include the OBU extension header.

Note: The change to the first requirement in the previous slide affects **layer-specific metadata OBUs**, because frame headers and tile group OBUs already need to satisfy that requirement.

6.7.1. General metadata OBU semantics

New paragraph before the **metadata_type** table.

Metadata OBUs may or may not have an OBU extension header. If there is no extension header, the metadata OBU's layer scope is all operating points in the coded video sequence, starting from the point the metadata OBU appears in the bitstream. If there is an extension header, the metadata OBU's layer scope is for the specific layer identified in the OBU extension header, starting from the point it appears in the bitstream. Specific metadata OBU types may have additional constraints on the presence and use of the OBU extension header. The metadata_type table below indicates which metadata OBU types are layer-specific and thus include an OBU extension header.

6.7.1. General metadata OBU semantics

The **metadata_type** table geta a new “Layer-specific” column.

New paragraph after the **metadata_type** table:

The semantics of the column “Layer-specific” and its values are defined in Section 6.2.2.

7.5. Ordering of OBUs

https://github.com/AOMediaCodec/av1-spec/pull/280

Before:

OBUs with spatial level IDs (spatial_id) greater than 0 must appear within a temporal unit in ***increasing*** order of the spatial level ID values.

After:

OBUs with obu_extension_flag equal to 1 must appear within a temporal unit in ***non-decreasing*** order of spatial_id values.

Backward compatibility: definition

Old decoders can decode new bitstreams

New decoders can decode old bitstreams

Backward compatibility: analysis

The drop_obu() condition is unchanged.

- Old and new decoders process and drop the same sets of OBUs.

New bitstreams need to satisfy more requirements on **obu_extension_flag**

- New bitstreams are a subset of old bitstreams
- Old decoders can decode new bitstreams

Can new decoders decode old bitstreams?

New decoders can decode old bitstreams

New decoders should not enforce the new requirements on obu_extension_flag

New decoders

- Should allow non-layer-specific OBUs to have obu_extension_flag equal to 1
- Should allow layer-specific OBUs of the base layer to have obu_extension_flag equal to 0 when scalability is being used