AV1 Codec ISO Media File Format Binding

v1.0.0, 7 September 2018

This version:
https://aomediacodec.github.io/av1-isobmff/

Issue Tracking:
GitHub

Editors:
Cyril Concolato (Netflix)
Tom Finegan (Google)

Abstract

This document specifies the storage format for [AV1] bitstreams in [ISOBMFF] tracks as well as in [CMAF] files.

Table of Contents

§ 1. Bitstream features overview

An AV1 bitstream is composed of a sequence of OBUs, grouped into Temporal Units.

OBUs are made of a 1 or 2 bytes header, identifying in particular the type of OBU, followed by an optional length field and by an optional payload, whose presence and content depend on the OBU type. Depending on its type, an OBU can carry configuration information, metadata, or coded video data.

> NOTE: Tile List OBUs defined in the [AV1] specification are not supported in the current version of this specification. A future version of the specification may do so.

Temporal Units are processed by a decoder in the order given by the bitstream. Each Temporal Unit is associated with a presentation time. Some Temporal Units may contain multiple frames to be decoded but only one is presented.

> NOTE: The AV1 specification defines scalability features, allowing frames from multiple layers or operating points to be present in a single temporal unit. This version of storage in ISOBMFF supports the simple storage of scalable streams in a single track, but does not specify advanced tools for handling multi-track support, layer extraction, or other scalability related use cases. A future version of the specification may do so.

Frames carried in Temporal Units may have coding dependencies on frames carried previously in the same Temporal Unit or in previous Temporal Units. Frames that can be decoded without dependencies to previous frames are of two categories: Key Frames and Intra-only Frames. Frames that cannot be decoded independently are of three categories: Inter Frames, Switch Frames, and frames with a show_existing_frame flag set to 1.

Key Frames with the show_frame flag set to 1 have the additional property that after decoding the Key Frame, all frames following the Key Frame in the bitstream can be decoded. They are called Random Access Points in [AV1].

Key Frames with the show_frame flag set to 0 are called Delayed Random Access Points. Delayed Random Access Points have the additional property that if a future Key Frame Dependent Recovery Point exists, all frames following that Key Frame Dependent Recovery Point can be decoded. A Key Frame Dependent Recovery Point is a frame with show_existing_frame set to 1 that refers to a previous Delayed Random Access Points.

§ 2. Basic Encapsulation Scheme

This section describes the basic data structures used to signal encapsulation of AV1 bitstreams in [ISOBMFF] containers.

§ 2.1. General Requirements & Brands

A file conformant to this specification satisfies the following:

- It SHALL conform to the normative requirements of [ISOBMFF]
- It SHALL have the ***‘av01’*** brand among the compatible brands array of the FileTypeBox
- It SHALL contain at least one track using an AV1SampleEntry
- It SHOULD indicate a structural ISOBMFF brand among the compatible brands array of the FileTypeBox, such as ‘iso6’
- It MAY indicate CMAF brands as specified in §3 CMAF AV1 track format
- It MAY indicate other brands not specified in this document provided that the associated requirements do not conflict with those given in this specification

Parsers SHALL support the structures required by the `'iso6'` brand and MAY support structures required by further ISOBMFF structural brands.

§ 2.2. AV1 Sample Entry

§ 2.2.1. Definition

```
Sample Entry Type: av01
Container:         Sample Description Box ('stsd')
Mandatory:         Yes
Quantity:          One or more.
```

§ 2.2.2. Description

The ***AV1SampleEntry*** sample entry identifies that the track contains AV1 Samples, and uses an AV1-CodecConfigurationBox.

§ 2.2.3. Syntax

```
class AV1SampleEntry extends VisualSampleEntry('av01') {
  AV1CodecConfigurationBox config;
}
```

§ 2.2.4. Semantics

The ***width*** and ***height*** fields of the VisualSampleEntry SHALL equal the values of max_frame_width_minus_1 + 1 and max_frame_height_minus_1 + 1 of the Sequence Header OBU applying to the samples associated with this sample entry.

The width and height in the TrackHeaderBox SHOULD equal, respectively, the maximum Render-Width, called MaxRenderWidth, and the maximum RenderHeight, called MaxRenderHeight, of all the frames associated with this sample entry. Additionally, if MaxRenderWidth and MaxRenderHeight values do not equal respectively the max_frame_width_minus_1 + 1 and max_frame_height_minus_1 + 1 values of the Sequence Header OBU, a PixelAspectRatioBox box SHALL be present in the sample entry and set such that

```
hSpacing / vSpacing = MaxRenderWidth * (max_frame_height_minus_1 + 1) /
                              ((max_frame_width_minus_1 + 1) * MaxRenderHeig
```

The ***compressorname*** field of the VisualSampleEntry is an informative name. It is formatted in a fixed 32-byte field, with the first byte set to the number of bytes to be displayed, followed by that number of bytes of displayable data, followed by padding to complete 32 bytes total (including the size byte). The value "\012AOM Coding" is RECOMMENDED; the first byte is a count of the remaining bytes, here represented by \012, which (being octal 12) is decimal 10, the number of bytes in the rest of the string.

NOTE: Parsers may ignore the value of the compressorname field. It is specified in this document simply for legacy and backwards compatibility reasons.

The ***config*** field SHALL contain an AV1CodecConfigurationBox that applies to the samples associated with this sample entry.

NOTE: Multiple instances of AV1SampleEntry may be required when the track contains samples requiring a AV1CodecConfigurationBox with different characteristics.

Optional boxes not specifically mentioned here can be present, in particular those indicated in the definition of the VisualSampleEntry in [ISOBMFF].

§ 2.3. AV1 Codec Configuration Box

§ 2.3.1. Definition

```
Box Type:  av1C
Container: AV1 Sample Entry ('av01')
Mandatory: Yes
Quantity:  Exactly One
```

§ 2.3.2. Description

The ***AV1CodecConfigurationBox*** contains decoder configuration information that SHALL be valid for every sample that references the sample entry.

§ 2.3.3. Syntax

```
class AV1CodecConfigurationBox extends Box('av1C'){
  AV1CodecConfigurationRecord av1Config;
}

aligned (8) class AV1CodecConfigurationRecord {
  unsigned int (1) marker = 1;
  unsigned int (7) version = 1;
  unsigned int (3) seq_profile;
  unsigned int (5) seq_level_idx_0;
  unsigned int (1) seq_tier_0;
  unsigned int (1) high_bitdepth;
  unsigned int (1) twelve_bit;
  unsigned int (1) monochrome;
  unsigned int (1) chroma_subsampling_x;
  unsigned int (1) chroma_subsampling_y;
  unsigned int (2) chroma_sample_position;
  unsigned int (3) reserved = 0;


  unsigned int (1) initial_presentation_delay_present;
  if (initial_presentation_delay_present) {
    unsigned int (4) initial_presentation_delay_minus_one;
  } else {
    unsigned int (4) reserved = 0;
  }

  unsigned int (8)[] configOBUs;
}
```

§ 2.3.4. Semantics

The ***marker*** field SHALL be set to 1.

> NOTE: The marker bit ensures that the bit pattern of the first byte of the AV1CodecConfiguration-Record cannot be mistaken for an OBU Header byte.

The ***version*** field indicates the version of the AV1CodecConfigurationRecord. The value SHALL be set to 1 for AV1CodecConfigurationRecord.

The ***seq_profile*** field indicates the AV1 profile and SHALL be equal to the seq_profile value from the Sequence Header OBU.

The ***seq_level_idx_0*** field indicates the value of seq_level_idx[0] found in the Sequence Header OBU and SHALL be equal to the value of seq_level_idx[0] in the Sequence Header OBU.

The ***seq_tier_0*** field indicates the value of seq_tier[0] found in the Sequence Header OBU and SHALL be equal to the value of seq_tier[0] in the Sequence Header OBU.

The ***high_bitdepth*** field indicates the value of the high_bitdepth flag from the Sequence Header OBU.

The ***twelve_bit*** field indicates the value of the twelve_bit flag from the Sequence Header OBU.

The ***monochrome*** field indicates the value of the mono_chrome flag from the Sequence Header OBU.

The ***chroma_subsampling_x*** field indicates the subsampling_x value from the Sequence Header OBU.

The ***chroma_subsampling_y*** field indicates the subsampling_y value from the Sequence Header OBU.

The ***chroma_sample_position*** field indicates the chroma_sample_position value from the Sequence Header OBU.

The ***initial_presentation_delay_present*** field indicates the presence of the initial_presentation_delay_minus_one field.

The ***initial_presentation_delay_minus_one*** field indicates the number of samples (minus one) that need to be decoded prior to starting the presentation of the first sample associated with this sample entry in order to guarantee that each sample will be decoded prior to its presentation time under the constraints of the first level value indicated by seq_level_idx in the Sequence Header OBU (in the configOBUs field or in the associated samples). More precisely, the following procedure SHALL not return any error:

- construct a hypothetical bitstream consisting of the OBUs carried in the sample entry followed by the OBUs carried in all the samples referring to that sample entry,
- set the first initial_display_delay_minus_1 field of each Sequence Header OBU to the number of frames minus one contained in the first initial_presentation_delay_minus_one + 1 samples,
- set the frame_presentation_time field of the frame header of each presentable frame such that it matches the presentation time difference between the sample carrying this frame and the previous sample (if it exists, 0 otherwise),
- apply the decoder model specified in [AV1] to this hypothetical bitstream using the first operating point. If `buffer_removal_time` information is present in bitstream for this operating point, the decoding schedule mode SHALL be applied, otherwise the resource availability mode SHALL be applied.

NOTE: With the above procedure, when smooth presentation can be guaranteed after decoding the first sample, initial_presentation_delay_minus_one is 0.

NOTE: Because the above procedure considers all OBUs in all samples associated with a sample entry, if these OBUS form multiple coded video sequences which would have different values of `initial_presentation_delay_minus_one` if considered separately, the sample entry would signal the larger value.

EXAMPLE 1
The difference between initial_presentation_delay_minus_one and initial_display_delay_minus_1 can be illustrated by considering the following example:

```
a b c d e f g h
```

where letters correspond to frames. Assume that `initial_display_delay_minus_1` is 2, i.e. that once frame `c` has been decoded, all other frames in the bitstream can be presented on time. If those frames were grouped into temporal units and samples as follows:

```
[a] [b c d] [e] [f] [g] [h]
```

`initial_presentation_delay_minus_one` would be 1 because it takes presentation of 2 samples to ensure that `c` is decoded. But if the frames were grouped as follows:

```
[a] [b] [c] [d e f] [g] [h]
```

`initial_presentation_delay_minus_one` would be 2 because it takes presentation of 3 samples to ensure that `c` is decoded.

The ***configOBUs*** field contains zero or more OBUs. Any OBU may be present provided that the following procedures produce compliant AV1 bitstreams:

- From any sync sample, an AV1 bitstream is formed by first outputting the OBUs contained in the AV1CodecConfigurationBox and then by outputing all OBUs in the samples themselves, in order, starting from the sync sample.
- From any sample marked with the AV1ForwardKeyFrameSampleGroupEntry, an AV1 bitstream is formed by first outputting the OBUs contained in the AV1CodecConfigurationBox and then by outputing all OBUs in the sample itself, then by outputting all OBUs in the samples, in order, starting from the sample at the distance indicated by the sample group.

Additionally, the configOBUs field SHALL contain at most one Sequence Header OBU and if present, it SHALL be the first OBU.

> NOTE: The configOBUs field is expected to contain only one Sequence Header OBU and zero or more Metadata OBUs when applicable to all the associated samples.

OBUs stored in the configOBUs field follow the open_bitstream_unit Low Overhead Bitstream Format syntax as specified in [AV1]. The flag obu_has_size_field SHALL be set to 1, indicating that the size of the OBU payload follows the header, and that it is coded using LEB128.

When a Sequence Header OBU is contained within the configOBUs of the AV1CodecConfigurationRecord, the values present in the Sequence Header OBU contained within configOBUs SHALL match the values of the AV1CodecConfigurationRecord.

The presentation times of AV1 samples are given by the ISOBMFF structures. The timing_info_present_flag in the Sequence Header OBU (in the configOBUs field or in the associated samples) SHOULD be set to 0. If set to 1, the timing_info structure of the Sequence Header OBU, the frame_presentation_time and buffer_removal_time fields of the Frame Header OBUs, if present, SHALL be ignored for the purpose of timed processing of the ISOBMFF file.

The sample entry SHOULD contain a 'colr' box with a colour_type set to 'nclx'. If present, the values of colour_primaries, transfer_characteristics, and matrix_coefficients SHALL match the values given in the Sequence Header OBU (in the configOBUs field or in the associated samples) if the color_description_present_flag is set to 1. Similarly, the full_range_flag in the 'colr' box shall match the color_range flag in the Sequence Header OBU. When configOBUs does not contain a Sequence Header OBU, this box with colour_type set to 'nclx' SHALL be present.

The CleanApertureBox 'clap' SHOULD not be present.

For sample entries corresponding to HDR content, the MasteringDisplayColourVolumeBox 'mdcv' and ContentLightLevelBox 'clli' SHOULD be present, and their values SHALL match the values of contained in the Metadata OBUs of type METADATA_TYPE_HDR_CLL and METADATA_TYPE_HDR_MDCV, if present (in the configOBUs or in the samples).

> NOTE: The MasteringDisplayColourVolumeBox 'mdcv' and ContentLightLevelBox 'clli' have identical syntax to the SMPTE2086MasteringDisplayMetadataBox 'SmDm' and ContentLightLevelBox 'CoLL', except that they are of type `Box` and not `FullBox`. However, the semantics of the MasteringDisplayColourVolumeBox and SMPTE2086MasteringDisplayMetadataBox have important differences and should not be confused.

Additional boxes may be provided at the end of the VisualSampleEntry as permitted by ISOBMFF, that may represent redundant or similar information to the one provided in some OBUs contained in the AV1CodecConfigurationBox. If the box definition does not indicate that its information overrides the OBU information, in case of conflict, the OBU information should be considered authoritative.

§ 2.4. AV1 Sample Format

For tracks using the AV1SampleEntry, an ***AV1 Sample*** has the following constraints:

- the sample data SHALL be a sequence of OBUs forming a Temporal Unit,
- each OBU SHALL follow the open_bitstream_unit Low Overhead Bitstream Format syntax as specified in [AV1]. Each OBU SHALL have the obu_has_size_field set to 1 except for the last OBU in the sample, for which obu_has_size_field MAY be set to 0, in which case it is assumed to fill the remainder of the sample,

NOTE: When extracting OBUs from an ISOBMFF file, and depending on the capabilities of the decoder processing these OBUs, ISOBMFF parsers MAY need to either: set the obu_has_size_field to 1 for some OBUs if not already set, add the size field in this case, and add Temporal Delimiter OBU; or use the length-delimited bitstream format as defined in Annex B of AV1. If encryption is used, similar operations MAY have to be done before or after decryption depending on the demuxer/decryptor/decoder architecture.

- OBU trailing bits SHOULD be limited to byte alignment and SHOULD not be used for padding,
- OBUs of type OBU_TEMPORAL_DELIMITER, OBU_PADDING, or OBU_REDUNDANT_FRAME_HEADER SHOULD NOT be used.
- OBUs of type OBU_TILE_LIST SHALL NOT be used.

If an AV1 Sample is signaled as a sync sample (in the SyncSampleBox or by setting sample_is_non_sync_sample to 0), it SHALL be a Random Access Point as defined in [AV1], i.e. satisfy the following constraints:

- Its first frame is a Key Frame that has show_frame flag set to 1,
- It contains a Sequence Header OBU before the first Frame Header OBU.

NOTE: Within this definition, a sync sample may contain additional frames that are not Key Frames. The fact that none of them is the first frame in the temporal unit ensures that they are decodable.

> NOTE: Other types of OBUs such as metadata OBUs could be present before the Sequence Header OBU.

Intra-only frames SHOULD be signaled using the sample_depends_on flag set to 2.

Delayed Random Access Points SHOULD be signaled using sample groups and the AV1ForwardKeyFrameSampleGroupEntry.

Switch Frames SHOULD be signaled using sample groups and the AV1SwitchFrameSampleGroupEntry.

Additionally, if a file contains multiple tracks that are alternative representations of the same content, in particular using Switch Frames, those tracks SHOULD be marked as belonging to the same alternate group and should use a track selection box with an appropriate attribute (e.g. ‘bitr’).

In tracks using the AV1SampleEntry, the ‘ctts’ box and composition offsets in movie fragments SHALL NOT be used. Similarly, the is_leading flag, if used, SHALL be set to 0 or 2.

When a temporal unit contains more than one frame, the sample corresponding to that temporal unit MAY be marked using the AV1MultiFrameSampleGroupEntry.

Metadata OBUs may be carried in sample data. In this case, the AV1MetadataSampleGroupEntry SHOULD be used. If the metadata OBUs are static for the entire set of samples associated with a given sample description entry, they SHOULD also be in the OBU array in the sample description entry.

Unless explicitely stated, the grouping_type_parameter is not defined for the SampleToGroupBox with grouping types defined in this specification.

§ 2.5. AV1 Forward Key Frame sample group entry

§ 2.5.1. Definition

```
Group Type: av1f
Container:  Sample Group Description Box ('sgpd')
Mandatory:  No
Quantity:   Zero or more.
```

§ 2.5.2. Description

The ***AV1ForwardKeyFrameSampleGroupEntry*** documents samples that contain a Delayed Random Access Point that are followed at a given distance in the bitstream by a Key Frame Dependent Recovery Point.

§ 2.5.3. Syntax

```
class AV1ForwardKeyFrameSampleGroupEntry extends VisualSampleGroupEntry('av1
  unsigned int(8) fwd_distance;
}
```

§ 2.5.4. Semantics

The ***fwd_distance*** field indicates the number of samples between this sample and the next sample containing the associated Key Frame Dependent Recovery Point. 0 means the next sample.

§ 2.6. AV1 Multi-Frame sample group entry

§ 2.6.1. Definition

```
Group Type: av1m
Container:  Sample Group Description Box ('sgpd')
Mandatory:  No
Quantity:   Zero or more.
```

§ 2.6.2. Description

The ***AV1MultiFrameSampleGroupEntry*** documents samples that contain multiple frames.

§ 2.6.3. Syntax

```
class AV1MultiFrameSampleGroupEntry extends VisualSampleGroupEntry('av1m') {
}
```

§ 2.7. AV1 Switch Frame sample group entry

§ 2.7.1. Definition

```
Group Type: av1s
Container:  Sample Group Description Box ('sgpd')
Mandatory:  No
Quantity:   Zero or more.
```

§ 2.7.2. Description

The ***AV1SwitchFrameSampleGroupEntry*** documents samples that start with a Switch Frame.

§ 2.7.3. Syntax

```
class AV1SwitchFrameSampleGroupEntry extends VisualSampleGroupEntry('av1s')
}
```

§ 2.8. AV1 Metadata sample group entry

§ 2.8.1. Definition

```
Group Type: av1M
Container:  Sample Group Description Box ('sgpd')
Mandatory:  No
Quantity:   Zero or more.
```

§ 2.8.2. Description

The ***AV1MetadataSampleGroupEntry*** documents samples that contain metadata OBUs. The `grouping_type_parameter` can be used to identify samples containing metadata OBUs of a given type. If no `grouping_type_parameter` is provided, the sample group entry identifies samples containing metadata OBUs for which the `metadata_type` is unknown.

§ 2.8.3. Syntax

```
class AV1MetadataSampleGroupEntry extends VisualSampleGroupEntry('av1M') {
}
```

For this sample group entry, the `grouping_type_parameter` syntax is as follows:

```
{
   unsigned int (8) metadata_type;
   unsigned int (24) metadata_specific_parameters;
}
```

§ 2.8.4. Semantics

metadata_type is a 8-bit field whose value is the value of the metadata_type field defined in [AV1], when it is equal or lower than 255. metadata_type values above 255 are not supported by this sample group.

metadata_specific_parameters provides an additional part of the `grouping_type_parameter`, which MAY be used to distinguish different sample groups having the same `metadata_type` but different sub-parameters. In this version of the specification, `metadata_specific_parameters` is only defined when `metadata_type` is set to `METADATA_TYPE_ITUT_T35` in which case its value SHALL be set to the first 24 bits of the `metadata_itut_t35` structure. For other types of metadata, its value SHOULD be set to 0. In all cases, when the `grouping_type_parameter` is used, readers processing sample groups SHALL use the entire value of `grouping_type_parameter`.

§ 3. CMAF AV1 track format

[CMAF] defines structural constraints on ISOBMFF files additional to [ISOBMFF] for the purpose of, for example, adaptive streaming or for protected files. Conformance to these structural constraints is signaled by the presence of the brand `cmfc` in the `FileTypeBox`.

If a CMAF Video Track uses the brand `av01`, it is called a ***CMAF AV1 Track*** and the following constraints, defining the CMAF Media Profile for AV1, apply:

- it SHALL use an AV1SampleEntry
- it MAY use multiple sample entries, and in that case the following values SHALL not change in the track:
  - `seq_profile`
  - `still_picture`
  - the first value of `seq_level_idx`
  - the first value of `seq_tier`
  - `color_config`

    - `initial_presentation_delay_minus_one`

When protected, CMAF AV1 Tracks SHALL use the signaling defined in [CMAF], which in turn relies on [CENC], with the provisions specified in §4 Common Encryption.

§ 4. Common Encryption

CMAF AV1 Tracks and non-segmented AV1 files MAY be protected. If protected, they SHALL conform to [CENC]. Only the `cenc` and `cbcs` protection schemes are supported.

When the protected scheme `cenc` is used, samples SHALL be protected using subsample encryption and SHALL NOT use pattern encryption.

When the protected scheme `cbcs` is used, samples SHALL be protected using subsample encryption and SHALL use pattern encryption. [CENC] recommends that the Pattern Block length be 10, i.e. `crypt_byte_block + skip_byte_block = 10`. All combinations such that this is true SHALL be supported, including `crypt_byte_block` = 10 and `skip_byte_block` = 0.

§ 4.1. General Subsample Encryption constraints

Within protected samples, the following constraints apply:

- Protected samples SHALL be exactly spanned by one or more contiguous subsamples.
    - An OBU MAY be spanned by one or more subsamples, especially when it has multiple ranges of protected data. However, as recommended in [CENC], writers SHOULD reduce the number of subsamples as possible. This can be achieved by using a subsample that spans multiple consecutive unprotected OBUs as well as the first unprotected and protected parts of the following protected OBU, if such protected OBU exists.
    - A large unprotected OBU whose data size is larger than the maximum size of a single BytesOfClearData field MAY be spanned by multiple subsamples with zero size BytesOfProtectedData.
- All OBU Headers and associated obu_size fields SHALL be unprotected.
- Temporal Delimiter OBUs, Sequence Header OBUs, and Frame Header OBUs (including within a Frame OBU), Redundant Frame Header OBUs and Padding OBUs SHALL be unprotected.
- Metadata OBUs MAY be protected.

- Tile Group OBUs and Frame OBUs SHALL be protected. Within Tile Group OBUs or Frame OBUs, the following applies:
    - A subsample SHALL be created for each tile.
    - BytesOfProtectedData SHALL be a multiple of 16 bytes.
    - BytesOfProtectedData SHALL end on the last byte of the `decode_tile` structure (including any trailing bits).
    - BytesOfProtectedData SHALL span all complete 16-byte blocks of the `decode_tile` structure (including any trailing bits).

> NOTE: As a result of the above, partial blocks are not used and it is possible that BytesOfProtectedData does not start at the first byte of the `decode_tile` structure, but some number of bytes following that.

- - All other parts of Tile Group OBUs and Frame OBUs SHALL be unprotected.

§ 4.2. Subsample Encryption Illustration

Figure #1 illustrates Subsample based encryption of AV1.

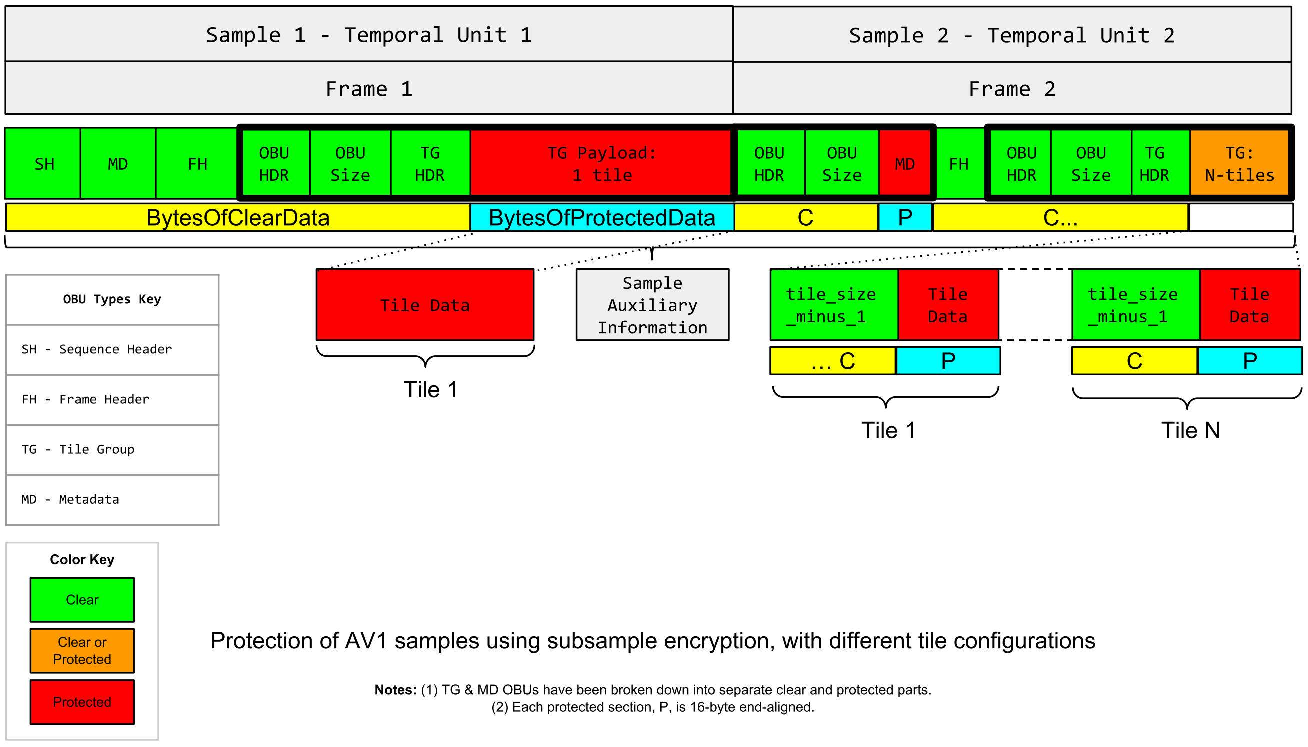

Figure 1 *Subsample-based AV1 encryption.*

§ 5. Codecs Parameter String

DASH and other applications require defined values for the ‘Codecs’ parameter specified in [RFC6381] for ISO Media tracks. The codecs parameter string for the AOM AV1 codec is as follows:

```
<sample entry 4CC>.<profile>.<level><tier>.<bitDepth>.<monochrome>.<chromaSu
<colorPrimaries>.<transferCharacteristics>.<matrixCoefficients>.<videoFullRa
```

All fields following the sample entry 4CC are expressed as double digit decimals, unless indicated otherwise. Leading or trailing zeros cannot be omitted.

The profile parameter value, represented by a single digit decimal, SHALL equal the value of seq_profile in the Sequence Header OBU.

The level parameter value SHALL equal the first level value indicated by seq_level_idx in the Sequence Header OBU.

The tier parameter value SHALL be equal to `M` when the first seq_tier value in the Sequence Header OBU is equal to 0, and `H` when it is equal to 1.

The bitDepth parameter value SHALL equal the value of BitDepth variable as defined in [AV1] derived from the Sequence Header OBU.

The monochrome parameter value, represented by a single digit decimal, SHALL equal the value of mono_chrome in the Sequence Header OBU.

The chromaSubsampling parameter value, represented by a three-digit decimal, SHALL have its first digit equal to subsampling_x and its second digit equal to subsampling_y. If both subsampling_x and subsampling_y are set to 1, then the third digit SHALL be equal to chroma_sample_position, otherwise it SHALL be set to 0.

The colorPrimaries, transferCharacteristics, matrixCoefficients, and videoFullRangeFlag parameter values SHALL equal the value of matching fields in the Sequence Header OBU, if color_description_present_flag is set to 1, otherwise they SHOULD not be set, defaulting to the values below. The videoFullRangeFlag is represented by a single digit.

For example, codecs="av01.0.04M.10.0.112.09.16.09.0" represents AV1 Main Profile, level 3.0, Main tier, 10-bit content, non-monochrome, with 4:2:0 chroma subsampling co-located with (0, 0) luma sample, ITU-R BT.2100 color primaries, ITU-R BT.2100 PQ transfer characteristics, ITU-R BT.2100 YCbCr color matrix, and studio swing representation.

The parameters sample entry 4CC, profile, level, tier, and bitDepth are all mandatory fields. If any of these fields are empty, or not within their allowed range, the processing device SHOULD treat it as an error.

All the other fields (including their leading '.') are optional, mutually inclusive (all or none) fields. If not specified then the values listed in the table below are assumed.

mono_chrome	0
chromaSubsampling	110 (4:2:0)
colorPrimaries	1 (ITU-R BT.709)
transferCharacteristics	1 (ITU-R BT.709)
matrixCoefficients	1 (ITU-R BT.709)
videoFullRangeFlag	0 (studio swing representation)

The string codecs="av01.0.01M.08" in this case would represent AV1 Main Profile, level 2.1, Main tier, 8-bit content with 4:2:0 chroma subsampling, ITU-R BT.709 color primaries, transfer characteristics, matrix coefficients, and studio swing representation.

If any character that is not '.', digits, part of the AV1 4CC, or a tier value is encountered, the string SHALL be interpreted ignoring all the characters starting from that character.

§ Conformance

Conformance requirements are expressed with a combination of descriptive assertions and RFC 2119 terminology. The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in the normative parts of this document are to be interpreted as described in RFC 2119. However, for readability, these words do not appear in all uppercase letters in this specification.

All of the text of this specification is normative except sections explicitly marked as non-normative, examples, and notes. [RFC2119]

Examples in this specification are introduced with the words “for example” or are set apart from the normative text with `class="example"`, like this:

EXAMPLE 2
This is an example of an informative example.

Informative notes begin with the word “Note” and are set apart from the normative text with `class="note"`, like this:

Note, this is an informative note.

§ Index

§ Terms defined by this specification

§ Terms defined by reference

[AV1] defines the following terms:

- av1 bitstream
- buffer_removal_time
- delayed random access point
- frame header obu
- frame obu
- frame_presentation_time
- initial_display_delay_minus_1
- inter frame
- intra-only frame
- key frame
- key frame dependent recovery point
- low overhead bitstream format
- max_frame_height_minus_1
- max_frame_width_minus_1
- metadata obu
- mono_chrome
- obu
- obu header
- obu_has_size_field
- obu_size
- open_bitstream_unit
- padding obu
- random access point
- redundant frame header obu
- seq_level_idx
- seq_tier
- sequence header obu
- show_existing_frame
- show_frame
- subsampling_x
- subsampling_y
- switch frame
- temporal delimiter obu
- temporal unit
- tile group obu
- tile list obu
- timing_info
- timing_info_present_flag

[CENC] defines the following terms:

- bytesofcleardata
- bytesofprotecteddata
- cbcs
- cenc

[CMAF] defines the following terms:

- cmaf video track
- cmfc

[ISOBMFF] defines the following terms:

- bitr
- clap
- clli
- colr
- ctts
- iso6
- mdcv
- nclx
- visualsampleentry

[RFC6381] defines the following terms:

- codecs

[vp9] defines the following terms:

- coll
- smdm

§ References

§ Normative References

[AV1]
AV1 Bitstream & Decoding Process Specification. Standard. URL: https://aomedia-codec.github.io/av1-spec/av1-spec.pdf

[CENC]
Information technology — MPEG systems technologies — Part 7: Common encryption in ISO base media file format files. Standard. URL: https://www.iso.org/standard/68042.html

[CMAF]
Information technology — Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media. Standard. URL: https://www.iso.org/standard/71975.html

[ISOBMFF]
Information technology — Coding of audio-visual objects — Part 12: ISO Base Media File Format. December 2015. International Standard. URL: http://standards.iso.org/ittf/PubliclyAvailableStandards/c068960_ISO_IEC_14496-12_2015.zip

[RFC2119]
S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. March 1997. Best Current Practice. URL: https://tools.ietf.org/html/rfc2119

[RFC6381]
R. Gellens; D. Singer; P. Frojdh. The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types. August 2011. Proposed Standard. URL: https://tools.ietf.org/html/rfc6381