INTERNATIONAL STANDARD

ISO/IEC 14496-15

First edition
2004-04-15

Information technology — Coding of audio-visual objects —

Part 15: Advanced Video Coding (AVC) file format

Technologies de l'information — Codage des objets audiovisuels —

Partie 15: Format de fichier de codage vidéo avancé (AVC)

Reference number
ISO/IEC 14496-15:2004(E)

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 14496-15 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

ISO/IEC 14496 consists of the following parts, under the general title *Information technology — Coding of audio-visual objects*:

— *Part 1: Systems*

— *Part 2: Visual*

— *Part 3: Audio*

— *Part 4: Conformance testing*

— *Part 5: Reference software*

— *Part 6: Delivery Multimedia Integration Framework (DMIF)*

— *Part 7: Optimized reference software for coding of audio-visual objects [Technical Report]*

— *Part 8: Carriage of ISO/IEC 14496 contents over IP networks*

— *Part 9: Reference hardware description [Technical Report]*

— *Part 10: Advanced Video Coding*

— *Part 11: Scene description and application engine*

— *Part 12: ISO base media file format*

— *Part 13: Intellectual Property Management and Protection (IPMP) extensions*

— *Part 14: MP4 file format*

— *Part 15: Advanced Video Coding (AVC) file format*

— *Part 16: Animation Framework eXtension (AFX)*

— *Part 17: Streaming text format*

— *Part 18: Font compression and streaming*

— *Part 19: Synthesized texture stream*

Introduction

The Advanced Video Coding (AVC) standard, jointly developed by the ITU-T and ISO/IEC SC29/WG11 (MPEG), offers not only increased coding efficiency and enhanced robustness, but also many features for the systems that use it. To enable the best visibility of, and access to, those features, and to enhance the opportunities for the interchange and interoperability of media, this part of ISO/IEC 14496 defines a storage format for video streams compressed using AVC.

This part of ISO/IEC 14496 defines a storage format based on, and compatible with, the ISO Base Media File Format (ISO/IEC 14496-12 and ISO/IEC 15444-12), which is used by the MP4 file format (ISO/IEC 14496-14) and the Motion JPEG 2000 file format (ISO/IEC 15444-3) among others. This part of ISO/IEC 14496 enables AVC video streams to:

- be used in conjunction with other media streams, such as audio;
- be used in an MPEG-4 systems environment, if desired;
- be formatted for delivery by a streaming server, using hint tracks;
- inherit all the use cases and features of the ISO Base Media File Format on which MP4 and MJ2 are based.

This part of ISO/IEC 14496 may be used as a standalone specification; it specifies how AVC content shall be stored in an ISO Base Media File Format compliant format. However, it is normally used in the context of a specification, such as the MP4 file format, derived from the ISO Base Media File Format, that permits the use of AVC video.

The ISO Base Media File Format is becoming increasingly common as a general-purpose media container format for the exchange of digital media, and its use in this context should accelerate both adoption and interoperability.

Extensions to the ISO Base Media File Format are defined here to support the new systems aspects of the AVC codec.

Information technology — Coding of audio-visual objects —

Part 15:
Advanced Video Coding (AVC) file format

1 Scope

This part of ISO/IEC 14496 specifies the storage format for AVC (ISO/IEC 14496-10 | ITU-T Rec. H.264) video streams.

The storage of AVC content uses the existing capabilities of the ISO Base Media File Format but also defines extensions to support the following features of the AVC codec:

- **Switching pictures**: To enable switching between different coded streams and substitution of pictures within the same stream.
- **Sub-sequences and layers**: Provides a structuring of the dependencies of a group of pictures to provide for a flexible stream structure (e.g. in terms of temporal scalability and layering).
- **Parameter sets**: The sequence and picture parameter set mechanism decouples the transmission of infrequently changing information from the transmission of coded macroblock data. Each slice containing the coded macroblock data references the picture parameter set containing its decoding parameters. In turn, the picture parameter set references a sequence parameter set that contains sequence level decoding parameter information.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14496-1:2001, *Information technology — Coding of audio-visual objects — Part 1: Systems*

ISO/IEC 14496-10, *Information technology — Coding of audio-visual objects — Part 10: Advanced video coding* | ITU-T Rec. H.264, *Advanced video coding for generic audiovisual services*

ISO/IEC 14496-12, *Information technology — Coding of audio-visual objects — Part 12: ISO base media file format* (technically identical to ISO/IEC 15444-12)

3 Terms, definitions, symbols and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 14496-1, ISO/IEC 14496-10 | ITU-T Rec. H.264 and the following apply.

3.1.1
parameter set
a sequence parameter set or a picture parameter set, as defined in ISO/IEC 14496-10

NOTE This term is used to refer to both types of parameter sets.

3.1.2
parameter set elementary stream
elementary stream containing samples made up of only sequence and picture parameter set NAL units synchronized with the video elementary stream

3.1.3
video elementary stream
elementary stream containing access units made up of NAL units for coded picture data

3.2 Symbols and abbreviated terms

AVC	Advanced Video Coding [ISO/IEC 14496-10]
HRD	Hypothetical Reference Decoder
IDR	Instantaneous Decoding Refresh
NAL	Network Abstraction Layer
PPS	Picture Parameter Set
SEI	Supplementary Enhancement Information
SPS	Sequence Parameter Set

4 Extensions to the ISO Base Media File Format

4.1 Introduction

This clause documents technical additions to the ISO Base Media File Format, which can be used when storing AVC streams. However, these additions could also be used by other media, if they are defined to use them. They are therefore documented here separately.

4.2 File identification

The brand ‘avc1’ shall be used to indicate that extensions conformant with this section are used in a file. The use of ‘avc1’ as a major-brand may be permitted by specifications; in that case, that specification defines the file extension and required behaviour.

4.3 Independent and Disposable Samples Box

Box Types: ‘sdtp’
Container: Sample Table Box (‘stbl’)
Mandatory: No
Quantity: Exactly one

This optional table answers three questions about sample dependency:

1) Does this sample depend on others (is it an I-picture)?

2) Do no other samples depend on this one?

3) Does this sample contain multiple (redundant) encodings of the data at this time-instant (possibly with different dependencies)?

In the absence of this table:

1) the sync sample table answers the first question; in most video codecs, I-pictures are also sync points,

2) the dependency of other samples on this one is unknown,

3) the existence of redundant coding is unknown.

When performing 'trick' modes, such as fast-forward, it is possible to use the first piece of information to locate independently decodable samples. Similarly, when performing random access, it may be necessary to locate the previous sync point or random access recovery point, and roll-forward from the sync point or the pre-roll starting point of the random access recovery point to the desired point. While rolling forward, samples on which no others depend need not be retrieved or decoded.

The value of 'sample-is-depended-on' is independent of the existence of redundant codings. However, a redundant coding may have different dependencies from the primary coding; if redundant codings are available, the value of 'sample-depends-on' documents only the primary coding.

The size of the table, `sample_count` is taken from the `sample_count` in the Sample Size Box ('stsz') or Compact Sample Size Box ('stz2').

4.3.1.1 Syntax

```
aligned(8) class SampleDependencyTypeBox
   extends FullBox('sdtp', version = 0, 0) {
   for (i=0; i < sample_count; i++){
      unsigned int(2) reserved = 0;
      unsigned int(2) sample-depends-on;
      unsigned int(2) sample-is-depended-on;
      unsigned int(2) sample-has-redundancy;
   }
}
```

4.3.1.2 Semantics

`sample-depends-on` takes one of the following four values:
- 0: the dependency of this sample is unknown;
- 1: this sample does depend on others (not an I picture);
- 2: this sample does not depend on others (I picture);
- 3: reserved.

`sample-is-depended-on` takes one of the following four values:
- 0: the dependency of other samples on this sample is unknown;
- 1: other samples depend on this one (not disposable);
- 2: no other sample depends on this one (disposable);
- 3: reserved.

`sample-has-redundancy` takes one of the following four values:
- 0: it is unknown whether there is redundant coding in this sample;
- 1: there is redundant coding in this sample;
- 2: there is no redundant coding in this sample;
- 3: reserved.

4.4 Sample groups

4.4.1 Introduction

This clause specifies a generic mechanism for representing a partition of the samples in a track. A *sample grouping* is an assignment of each sample in a track to be a member of one *sample group,* based on a

grouping criterion. A sample group in a sample grouping is not limited to being contiguous samples and may contain non-adjacent samples. As there may be more than one sample grouping for the samples in a track, each sample grouping has a type field to indicate the type of grouping. For example, a file might contain two sample groupings for the same track: one based on an assignment of sample to layers and another to sub-sequences.

Sample groupings are represented by two linked data structures: (1) a `SampleToGroup` box represents the assignment of samples to sample groups; (2) a `SampleGroupDescription` box contains a *sample group entry* for each sample group describing the properties of the group. There may be multiple instances of the `SampleToGroup` and `SampleGroupDescription` boxes based on different grouping criteria. These are distinguished by a type field used to indicate the type of grouping.

One example of using these tables is to represent the assignments of samples to *layers.* In this case each sample group represents one layer, with an instance of the `SampleToGroup` box describing which layer a sample belongs to. For more details, please refer to 5.3.12

4.4.2 SampleToGroup Box

4.4.2.1 Definition

Box Type: 'sbgp'
Container: Sample Table Box ('stbl')
Mandatory: No
Quantity: Zero or more.

This table can be used to find the group that a sample belongs to and the associated description of that sample group. The table is compactly coded with each entry giving the index of the first sample of a run of samples with the same sample group descriptor. The sample group description ID is an index that refers to a `SampleGroupDescription` box, which contains entries describing the characteristics of each sample group.

There may be multiple instances of this box if there is more than one sample grouping for the samples in a track. Each instance of the `SampleToGroup` box has a type code that distinguishes different sample groupings. Within a track, there shall be at most one instance of this box with a particular grouping type. The associated `SampleGroupDescription` shall indicate the same value for the grouping type.

4.4.2.2 Syntax

```
aligned(8) class SampleToGroupBox
   extends FullBox('sbgp', version = 0, 0)
{
   unsigned int(32)  grouping_type;
   unsigned int(32)  entry_count;
   for (i=1; i <= entry_count; i++)
   {
      unsigned int(32)  sample_count;
      unsigned int(32)  group_description_index;
   }
}
```

4.4.2.3 Semantics

`version` is an integer that specifies the version of this box.

`grouping_type` is an integer that identifies the type (i.e. criterion used to form the sample groups) of the sample grouping and links it to its sample group description table with the same value for grouping type. At most one occurrence of this box with the same value for `grouping_type` shall exist for a track.

`entry_count` is an integer that gives the number of entries in the following table.

`sample_count` is an integer that gives the number of consecutive samples with the same sample group descriptor.

`group_description_index` is an integer that gives the index of the sample group entry which describes the samples in this group. The index ranges from 1 to the number of sample group entries in the `SampleGroupDescription` Box, or takes the value 0 to indicate that this sample is a member of no group of this type.

4.4.3 SampleGroupDescription Box

4.4.3.1 Definition

Box Types: 'sgpd'
Container: Sample Table Box ('stbl')
Mandatory: No
Quantity: Zero or more, with one for each SampleToGroup Box.

This description table gives information about the characteristics of sample groups. The descriptive information is any other information needed to define or characterize the sample group.

There may be multiple instances of this box if there is more than one sample grouping for the samples in a track. Each instance of the `SampleGroupDescription` box has a type code that distinguishes different sample groupings. Within a track, there shall be at most one instance of this box with a particular grouping type. The associated `SampleToGroup` shall indicate the same value for the grouping type.

The information is stored in the sample group description box after the entry-count. An abstract entry type is defined and sample groupings shall define derived types to represent the description of each sample group. For video tracks, an abstract `VisualSampleGroupEntry` is used with similar types for audio and hint tracks.

4.4.3.2 Syntax

```
// Sequence Entry
abstract class SampleGroupDescriptionEntry (unsigned int(32) handler_type)
{
}



// Visual Sequence
abstract class VisualSampleGroupEntry (type) extends SampleGroupDescriptionEntry
(type)
{
}



// Audio Sequences
abstract class AudioSampleGroupEntry (type) extends SampleGroupDescriptionEntry
(type)
{
}
```

```
aligned(8) class SampleGroupDescriptionBox (unsigned int(32) handler_type)
   extends FullBox('sgpd', 0, 0){
   unsigned int(32) grouping_type;
   unsigned int(32) entry_count;
   int i;
   for (i = 1 ; i <= entry_count ; i++){
      switch (handler_type){
         case ‘vide’: // for video tracks
            VisualSampleGroupEntry ();
            break;
         case ‘soun’: // for audio tracks
            AudioSampleGroupEntry();
            break;
         case ‘hint’: // for hint tracks
            HintSampleGroupEntry();
            break;
      }
   }
}
```

4.4.3.3 Semantics

`version` is an integer that specifies the version of this box.

`grouping_type` is an integer that identifies the `SampleToGroup` box that is associated with this sample group description.

`entry_count` is an integer that gives the number of entries in the following table.

4.5 Random access recovery points

In some coding systems it is possible to random access into a stream and achieve correct decoding after having decoded a number of samples. This is known as gradual decoding refresh. For example, in video, the encoder might encode intra-coded macroblocks in the stream, such that it knows that within a certain period the entire picture consists of pixels that are only dependent on intra-coded macroblocks supplied during that period.

Samples for which such gradual refresh is possible are marked by being a member of this group. The definition of the group allows the marking to occur at either the beginning of the period or the end. However, when used with a particular media type, the usage of this group may be restricted to marking only one end (i.e. restricted to only positive or negative roll values). A roll-group is defined as that group of samples having the same roll distance.

4.5.1 Syntax

```
class VisualRollRecoveryEntry() extends VisualSampleGroupEntry (’roll’)
{
   signed int(16) roll-distance;
}
```

4.5.2 Semantics

`roll-distance` is a signed integer that gives the number of samples that must be decoded in order for a sample to be decoded correctly. A positive value indicates the number of samples after the sample that is a group member that must be decoded before recovery is complete. A negative value indicates the number of samples before the sample that is a group member that must be decoded in order for recovery to be complete at the marked sample. The value zero must not be used; the sync sample table documents random access points for which no recovery roll is needed.

4.6 Representation of new structures in movie fragments

Support for new SampleGroup structures within movie fragments is provided by the use of the `SampleToGroup` Box with the container for this Box being the `Track Fragment` Box ('traf'). The definition, syntax and semantics of this Box is as specified in 4.4.2.

The `SampleToGroup` Box can be used to find the group that a sample in a track fragment belongs to and the associated description of that sample group. The table is compactly coded with each entry giving the index of the first sample of a run of samples with the same sample group descriptor. The sample group description ID is an index that refers to a `SampleGroupDescription` Box, which contains entries describing the characteristics of each sample group and present in the `SampleTableBox`.

There may be multiple instances of the `SampleToGroup` Box if there is more than one sample grouping for the samples in a track fragment. Each instance of the `SampleToGroup` Box has a type code that distinguishes different sample groupings. The associated `SampleGroupDescription` shall indicate the same value for the grouping type.

To provide for further possible compaction, default sample groupings can be provided at a global level (per track), within the `Track Extends` Box. There can be multiple instances of the `SampleToGroup` Box within the `Track Extends` Box, if there are more than one default sample groupings for the samples in a track fragment. The presence of a `SampleToGroup` Box, within a `Track Fragment` Box, overrides the default values provided at the global level for that particular fragment.

The total number of samples represented in any `SampleToGroup` Box in the track fragment must match the total number of samples in all the track fragment runs. Each `SampleToGroup` Box documents a different grouping of the same samples.

A sample dependency Box may also occur in the track fragment Box. The 12-bit reserved field in movie fragments, as documented in ISO/IEC 14496-12, 8.31.1, is re-defined to include the sample-dependency-type information as defined above in 4.3, with a preceding 6-bit reserved field.

5 AVC elementary streams and sample definitions

This clause specifies the elementary stream and sample structure used to store AVC visual content inside the AVC file format.

5.1 Elementary stream structure

AVC specifies a set of Network Abstraction Layer (NAL) units, which contain different types of data. This subclause specifies the format of the elementary streams for storing such AVC content within the AVC file format. Two types of elementary streams are defined for this purpose (see also Figure 1):

- **Video elementary streams** shall contain all video coding related NAL units (i.e. those NAL units containing video data or signalling video structure) and may contain non-video coding related NAL units such as SEI message and access unit delimiter NAL units.

- **Parameter set elementary streams** shall contain only sequence and picture parameter set NAL units.

Using these stream types, AVC content shall be stored in either one or two elementary streams:

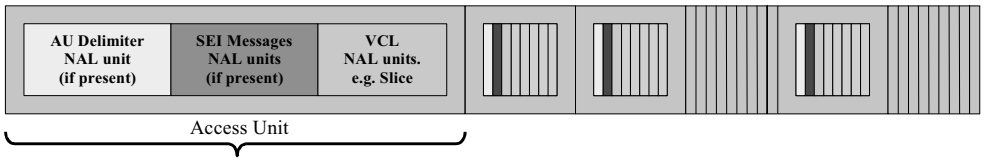
- **Video elementary stream only**: In this case, sequence and picture parameter set NAL units shall be stored in the sample descriptions of this track. Sequence and picture parameter set NAL units shall not be part of AVC samples within the stream itself.

- **Video elementary stream and parameter set elementary stream**: In this case, sequence and picture parameter set NAL units shall be transmitted only in the parameter set elementary stream and shall neither be present in the sample descriptions nor the AVC samples of the video elementary stream.

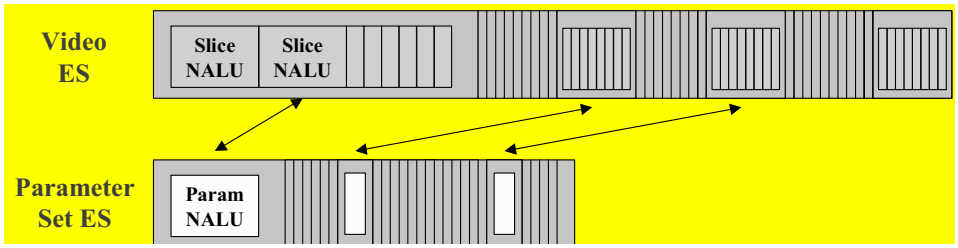
The types of NAL units that are allowed in each of the video and parameter set elementary streams are specified in Table 1.

Table 1 — NAL unit types in elementary streams

Value of nal_unit_type	Description	Video elementary stream	Parameter set elementary stream
0	Unspecified	Not specified by this part of ISO/IEC 14496	Not specified by this part of ISO/IEC 14496
1	Coded slice of a non-IDR picture slice_layer_without_partitioning_rbsp( )	Yes	No
2	Coded slice data partition A slice_data_partition_a_layer_rbsp( )	Yes	No
3	Coded slice data partition B slice_data_partition_b_layer_rbsp( )	Yes	No
4	Coded slice data partition C slice_data_partition_c_layer_rbsp( )	Yes	No
5	Coded slice of an IDR picture slice_layer_without_partitioning_rbsp( )	Yes	No
6	Supplemental enhancement information(SEI) sei_rbsp( )	Yes. Except for the Sub-sequence, layering or Filler SEI messages	No
7	Sequence parameter set (SPS) seq_parameter_set_rbsp( )	No. If parameter set elementary stream is not used, SPS shall be stored in the Decoder Specific Information.	Yes
8	Picture parameter set (PPS) pic_parameter_set_rbsp( )	No. If parameter set elementary stream is not used, PPS shall be stored in the Decoder Specific Information.	Yes
9	Access unit delimiter (AU Delimiter) access_unit_delimiter_rbsp( )	Yes	No
10	End of sequence end_of_seq_rbsp()	Yes	No
11	End of stream end_of_stream_rbsp()	Yes	No
12	Filler data (FD) filler_data_rbsp( )	No	No
13 – 23	Reserved	No	No
24 – 31	Unspecified	Not specified by this part of ISO/IEC 14496	Not specified by this part of ISO/IEC 14496



(a) Single video elementary stream containing NAL units



(b) Synchronized video and parameter sets with arrows denoting synchronization between streams

Figure 1 — AVC elementary stream structure

5.2 Sample and Configuration definition

5.2.1 Introduction

AVC sample: An AVC sample is an access unit as defined in ISO/IEC 14496-10, 7.4.1.2.

AVC parameter set sample: An AVC parameter set sample is a sample in a parameter set stream which shall consist of those parameter set NAL units that are to be considered as if present in the video elementary stream at the same instant in time.

5.2.2 Canonical order and restrictions

The AVC elementary stream is stored in the ISO Base Media File Format in a *canonical* format. The canonical format is as *neutral* as possible so that systems that need to customize the stream for delivery over different transport protocols — MPEG-2 Systems, RTP, and so on — should not have to *remove* information from the stream while being free to *add* to the stream. Furthermore, a canonical format allows such operations to be performed against a known initial state.

The canonical stream format is an AVC elementary stream that satisfies the following conditions:

- **Video data NAL units (Coded Slice, Coded Slice Data Partition A, Coded Slice Data Partition B, Coded Slice Data Partition C, Coded Slice IDR Pictures)**: All slice and data partition NAL units for a single picture shall be contained with the sample whose decoding time and composition time are those of the picture. Each AVC sample shall contain at least one video data NAL unit of the primary picture.

- **SEI message NAL units**: All SEI message NAL units shall be contained in the sample whose decoding time is that before which the SEI messages come into effect instantaneously. The order of SEI messages within a sample is as defined in ISO/IEC 14496-10, 7.4.1.2. This means that the SEI messages for a picture shall be included in the sample containing that picture and that SEI messages pertaining to a sequence of pictures shall be included in the sample containing the first picture of the sequence to which the SEI message pertains.

- **Access unit delimiter NAL units**: The constraints obeyed by access unit delimiter NAL units are defined in ISO/IEC 14496-10, 7.4.1.2.3.

- **Parameter sets***:* If a parameter set elementary stream is used, then the sample in the parameter stream shall have a decoding time equal or prior to when the parameter set(s) comes into effect

instantaneously. This means that for a parameter set to be used in a picture it must be sent prior to the sample containing that picture or in the sample for that picture.

NOTE Parameter sets are stored either in the sample descriptions of the video stream or in the parameter set stream, but never in both. This ensures that it is not necessary to examine every part of the video elementary stream to find relevant parameter sets. It also avoids dependencies of indefinite duration between the sample that contains the parameter set definition and the samples that use it. Storing parameter sets in the sample descriptions of a video stream provides a simple and static way to supply parameter sets. Parameter set elementary streams on the other hand are more complex but allow for more dynamism in the case of updates. Parameter sets may be inserted into the video elementary stream when the file is streamed over a transport that permits such parameter set updates.

- The sequence of NAL units in an elementary stream and within a single sample must be in a valid decoding order for those NAL units as specified in ISO/IEC 14496-10.

- **Parameter set track:** A sync sample in a parameter set track indicates that all parameter sets needed from that time forward in the video elementary stream are in that or succeeding parameter stream samples. Also there shall be a parameter set sample at each point a parameter set is updated. Each parameter set sample shall contain exactly the sequence and picture parameter sets needed to decode the relevant section of the video elementary stream.

  NOTE The use of a parameter set track in the file format does not require that a system delivering AVC content use a separate elementary stream for parameter sets. Instead, implementations may choose to map parameter sets to in-band parameter set NAL units in the video elementary stream or use some out-of-band delivery mechanism defined by the transport layer.

- **All timing information is external to stream.** Picture Timing SEI messages that define presentation or composition timestamps may be included in the AVC video elementary stream, as this message contains other information than timing, and may be required for conformance checking. However, all timing information is provided by the information stored in the various sample metadata tables, and this information over-rides any timing provided in the AVC layer. Timing provided within the AVC stream in this file format should be ignored as it may contradict the timing provided by the file format and may not be correct or consistent within itself.

  NOTE This constraint is imposed due to the fact that post-compression editing, combination, or re-timing of a stream at the file format level may invalidate or make inconsistent any embedded timing information present within the AVC stream.

- **Sub-sequence and layering SEI messages.** Sub-sequence or layering SEI messages shall not occur in the AVC elementary stream. Specifically, the sub-sequence information, sub-sequence layer characteristics, and sub-sequence characteristics SEI messages shall not occur in the stored AVC video elementary stream. Instead, all such information is stored as external metadata as described in 5.3.12.

- **Redundant picture:** NAL units within a single access unit shall be ordered in non-decreasing order of redundant picture count (redundant_pic_cnt).

- **Slice groups:** NAL units within a primary coded picture or a redundant coded picture shall be ordered in non-decreasing order of slice group identifier. Within the same slice group, slices shall be ordered by their first Macroblock location (first_mb_in_slice in the slice header).

  NOTE Slice groups are stored in a canonical order to ease hinting, and to make it easier to find a primary picture within a sample.

- **No start codes.** The elementary streams shall not include start codes. As stored, each NAL unit is preceded by a length field as specified in 5.2.3; this enables easy scanning of the sample's NAL units. Systems that wish to deliver, from this file format, a stream using start codes will need to reformat the stream to insert those start codes.

- **No filler data.** Video data is naturally represented as variable bit rate in the file format and should be filled for transmission if needed. Filler Data NAL units and Filler Data SEI messages shall not be present in the file format stored stream.

  NOTE The removal of Filler Data NAL units, start codes, zero_byte syntax elements, SEI messages or Filler Data SEI messages may change the bit-stream characteristics with respect to conformance with the HRD when operating the HRD in CBR mode as specified in ISO/IEC 14496-10, Annex C.

5.2.3 AVC sample structure definition

This subclause defines structure for the samples of AVC streams. Samples are externally framed and have a size supplied by that external framing. An example of the structure of an AVC sample is depicted in Figure 2.

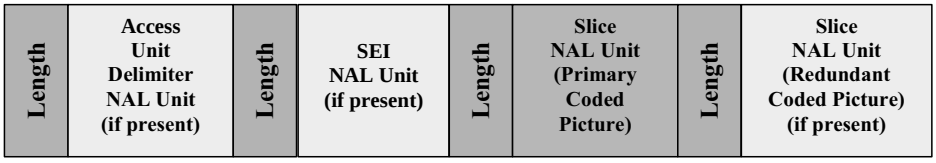


Figure 2 — The structure of an AVC sample

An AVC access unit is made up of a set of NAL units. Each NAL unit is represented with a:

- *Length*: Indicates the length in bytes of the following NAL unit. The length field can be configured to be of 1, 2, or 4 bytes.

- *NAL Unit*: Contains the NAL unit data as specified in ISO/IEC 14496-10.

5.2.4 Decoder configuration information

This subclause specifies the decoder configuration information for ISO/IEC 14496-10 video content.

5.2.4.1 AVC decoder configuration record

This record contains the size of the length field used in each sample to indicate the length of its contained NAL units as well as the initial parameter sets. This record is externally framed (its size must be supplied by the structure which contains it).

This record contains a version field. This version of the specification defines version 1 of this record. Incompatible changes to the record will be indicated by a change of version number. Readers must not attempt to decode this record or the streams to which it applies if the version number is unrecognised.

Compatible extensions to this record will extend it and will not change the configuration version code. Readers should be prepared to ignore unrecognised data beyond the definition of the data they understand (e.g. after the parameter sets in this specification).

When used to provide the configuration of

- a parameter set elementary stream,

- a video elementary stream used in conjunction with a parameter set elementary stream,

the configuration record shall contain no sequence or picture parameter sets (`numOfSequenceParameterSets` and `numOfPictureParameterSets` shall both have the value 0).

The values for AVCProfileIndication, AVCLevelIndication, and the flags which indicate profile compatibility must be valid for all parameter sets of the stream described by this record. The level indication must indicate a level of capability equal to or greater than the highest level indicated in the included parameter sets; each profile compatibility flag may only be set if all the included parameter sets set that flag. The profile indication

must indicate a profile to which the entire stream conforms. If the sequence parameter sets are marked with different profiles, and the relevant profile compatibility flags are all zero, then the stream may need examination to determine which profile, if any, the stream conforms to. If the stream is not examined, or the examination reveals that there is no profile to which the stream conforms, then the stream must be split into two or more sub-streams with separate configuration records in which these rules can be met.

5.2.4.1.1 Syntax

```
aligned(8) class AVCDecoderConfigurationRecord {
   unsigned int(8) configurationVersion = 1;
   unsigned int(8) AVCProfileIndication;
   unsigned int(8) profile_compatibility;
   unsigned int(8) AVCLevelIndication;
   bit(6) reserved = '111111'b;
   unsigned int(2) lengthSizeMinusOne;
   bit(3) reserved = '111'b;
   unsigned int(5) numOfSequenceParameterSets;
   for (i=0; i< numOfSequenceParameterSets;  i++) {
      unsigned int(16) sequenceParameterSetLength ;
      bit(8*sequenceParameterSetLength) sequenceParameterSetNALUnit;
   }
   unsigned int(8) numOfPictureParameterSets;
   for (i=0; i< numOfPictureParameterSets;  i++) {
      unsigned int(16) pictureParameterSetLength;
      bit(8*pictureParameterSetLength) pictureParameterSetNALUnit;
   }
}
```

5.2.4.1.2 Semantics

`AVCProfileIndication` contains the profile code as defined in ISO/IEC 14496-10.

`profile_compatibility` is a byte defined exactly the same as the byte which occurs between the profile_IDC and level_IDC in a sequence parameter set (SPS), as defined in ISO/IEC 14496-10.

`AVCLevelIndication` contains the level code as defined in ISO/IEC 14496-10.

`lengthSizeMinusOne` indicates the length in bytes of the `NALUnitLength` field in an AVC video sample or AVC parameter set sample of the associated stream minus one. For example, a size of one byte is indicated with a value of 0. The value of this field shall be one of 0, 1, or 3 corresponding to a length encoded with 1, 2, or 4 bytes, respectively.

`numOfSequenceParameterSets` indicates the number of SPSs that are used as the initial set of SPSs for decoding the AVC elementary stream.

`sequenceParameterSetLength` indicates the length in bytes of the SPS NAL unit as defined in ISO/IEC 14496-10.

`sequenceParameterSetNALUnit` contains a SPS NAL unit, as specified in ISO/IEC 14496-10. SPSs shall occur in order of ascending parameter set identifier with gaps being allowed.

`numOfPictureParameterSets` indicates the number of picture parameter sets (PPSs) that are used as the initial set of PPSs for decoding the AVC elementary stream.

`pictureParameterSetLength` indicates the length in bytes of the PPS NAL unit as defined in ISO/IEC 14496-10.

`pictureParameterSetNALUnit` contains a PPS NAL unit, as specified in ISO/IEC 14496-10. PPSs shall occur in order of ascending parameter set identifier with gaps being allowed.

5.3 Derivation from ISO Base Media File Format

5.3.1 Introduction

This subclause defines how the AVC file format is derived from the ISO Base Media File Format [ISO/IEC 14496-12].

Table 2 summarizes the correspondences between the sets of terminology used in AVC Video and the ISO Base Media File Format.

Table 2 — Correspondence of terms in AVC and ISO Base Media File Format

AVC	ISO Base Media File Format
-	Movie
Stream	Track
Access Unit	Sample

5.3.2 AVC File type and identification

Conformance with this part of ISO/IEC 14496 is indicated by the presence of the brand of a specification that permits the inclusion of AVC content, in the compatible brands list of the `FileTypeBox` as defined in ISO/IEC 14496-12. The file extension normally matches the major brand.

AVC content may be used in an MPEG-4 context. If the extensions documented in Clause 4 are not used or their support is not required in the decoder, then the brands ‘mp41’ or ‘mp42’ may be used. If the extensions are required, then the brand ‘avc1’ should be used. In this case, in a file with extension “.mp4”, the major brand may be ‘avc1’.

Readers conformant to this part of ISO/IEC 14496 should read the file if a suitable brand occurs in the compatible-brands list. Other structures and/or track types, defined in specifications other than that identified by the brand, may be present, and these may be ignored by a reader conformant with the specification identified by the brand.

5.3.3 AVC Track Structure

In the terminology of ISO/IEC 14496-12, AVC tracks (both video and parameter set tracks) are video or visual tracks. They therefore use:

a) a handler_type of ‘vide’ in the HandlerBox;

b) a video media header `'vmhd'`;

c) and, as defined below, a derivative of the `VisualSampleEntry`.

5.3.4 AVC Video Stream Definition

This subclause defines the sample entry and sample format for AVC video elementary streams.

5.3.4.1 Sample description name and format

5.3.4.1.1 Definition

Box Types: ‘avc1’, ‘avcC’, ‘m4ds’,’btrt’
Container: Sample Table Box (‘stbl’)
Mandatory: The avc1 box is mandatory
Quantity: One or more sample entries may be present

An AVC visual sample entry shall contain an AVC Configuration Box, as defined below. This includes an `AVCDecoderConfigurationRecord`, as defined in 5.2.4.1.

An optional `MPEG4BitRateBox` may be present in the AVC visual sample entry to signal the bit rate information of the AVC video stream. Extension descriptors that should be inserted into the Elementary Stream Descriptor, when used in MPEG-4, may also be present.

Multiple sample descriptions may be used, as permitted by the ISO Base Media File Format specification, to indicate sections of video that use different configurations or parameter sets.

5.3.4.1.2 Syntax

```
   // Visual Sequences
class AVCConfigurationBox extends Box(‘avcC’) {
   AVCDecoderConfigurationRecord() AVCConfig;
}


class MPEG4BitRateBox extends Box(‘btrt’){
   unsigned int(32) bufferSizeDB;
   unsigned int(32) maxBitrate;
   unsigned int(32) avgBitrate;
}

class MPEG4ExtensionDescriptorsBox extends Box(‘m4ds’) {
   Descriptor Descr[0 .. 255];
}

class AVCSampleEntry() extends VisualSampleEntry (‘avc1’){
   AVCConfigurationBox  config;
   MPEG4BitRateBox ();              // optional
   MPEG4ExtensionDescriptorsBox (); // optional
}
```

5.3.4.1.3 Semantics

`Compressorname` in the base class `VisualSampleEntry` indicates the name of the compressor used with the value `"\012AVC Coding"` being recommended (\012 is 10, the length of the string as a byte).

`config` is defined in 5.2.4. If a separate parameter set stream is used, numOfSequenceParameterSets and numOfPictureParameterSets must both be zero.

`Descr` is a descriptor which should be placed in the `ElementaryStreamDescriptor` when this stream is used in an MPEG-4 systems context. This does not include `SLConfigDescriptor` or `DecoderConfigDescriptor`, but includes the other descriptors in order to be placed after the `SLConfigDescriptor`.

`bufferSizeDB` gives the size of the decoding buffer for the elementary stream in bytes.

`maxBitrate` gives the maximum rate in bits/second over any window of one second.

`avgBitrate` gives the average rate in bits/second over the entire presentation.

5.3.4.2 Sample format

This subclause defines the format of a sample in an AVC video elementary stream. The syntax of a sample is configured via the decoder specific configuration for the AVC elementary stream. If the coded representations of a picture contain redundant pictures, then the constraints obeyed by the order of the coded pictures are as defined in ISO/IEC 14496-10, 7.4.1.2.3.

5.3.4.2.1 Syntax

```
aligned(8) class AVCSample
{
   unsigned int PictureLength = sample_size; //Size of AVCSample from
SampleSizeBox
   for (i=0; i<PictureLength; )     // to end of the picture
   {
      unsigned int((AVCDecoderConfigurationRecord.LengthSizeMinusOne+1)*8)
         NALUnitLength;
      bit(NALUnitLength * 8) NALUnit;
      i += (AVCDecoderConfigurationRecord.LengthSizeMinusOne+1) + NALUnitLength;
   }
}
```

5.3.4.2.2 Semantics

`NALUnitLength` indicates the size of a NAL unit measured in bytes. The length field includes the size of both the one byte NAL header and the EBSP payload but does not include the length field itself.

`NALUnit` contains a single NAL unit. The syntax of a NAL unit is defined in ISO/IEC 14496-10 and includes both the one byte NAL header and the variable length encapsulated byte stream payload.

5.3.5 AVC parameter set stream definition

This subclause defines the sample entry and sample format for AVC parameter set streams.

5.3.5.1 Sample description name and format

5.3.5.1.1 Definition

Box Types: ‘avcp’
Container: Sample Table Box (‘stbl’)
Mandatory: Yes
Quantity: One or more sample entries may be present

An AVC parameter stream sample entry shall contain an AVC Parameter Stream Configuration Box, as defined below.

5.3.5.1.2 Syntax

```
class AVCParameterSampleEntry() extends VisualSampleEntry (‘avcp’){
   AVCConfigurationBox  config;
}
```

5.3.5.1.3 Semantics

`Compressorname` in the base class `VisualSampleEntry` indicates the name of the compressor used with the value `"\016AVC Parameters"` being recommended (\016 is 14, the length of the string as a byte).

`config` is defined in 5.2.4. `numOfSequenceParameterSets` and `numOfPictureParameterSets` must both be zero.

5.3.5.2 Sample format

This subclause defines the sample format for AVC Parameter set streams. An AVC parameter set sample contains only one or more sequence or picture parameter set NAL units.

5.3.5.2.1 Syntax

```
aligned(8) class AVCParameterSample
{
   unsigned int PictureLength = sample_size;
            //Size of AVCParameterSample from SampleSizeBox
   for (i=0; i<PictureLength; )     // to end of the picture
   {
      unsigned int((AVCDecoderConfigurationRecord.LengthSizeMinusOne+1)*8)
         NALUnitLength;
      bit(NALUnitLength * 8) NALUnit;
      i += (AVCDecoderConfigurationRecord.LengthSizeMinusOne+1) + NALUnitLength;
   }
}
```

5.3.5.2.2 Semantics

`NALUnitLength` indicates the size of a NAL unit measured in bytes. The length field includes the size of both the one byte NAL header and the EBSP payload but does not include the length field itself.

`NALUnit` contains a single NAL unit. The syntax of a NAL unit is defined in ISO/IEC 14496-10 and includes both the one byte NAL header and the variable length encapsulated byte stream payload.

5.3.5.3 Track reference

A track reference of type ‘avcp’ in the video elementary stream track reference table, referencing the parameter set stream, is used to connect from the video elementary stream to the parameter set elementary stream.

5.3.6 Template fields used

The ISO Base Media File Format defines a number of fields which have default values but which may be defined for use by specific sub-systems. Tracks containing AVC data use the following template fields:

a) `alternate_group` in the `TrackHeaderBox` (see 5.3.13 on stream switching).

5.3.7 Visual width and height

The width and height fields in a `VisualSampleEntry` must correctly document the cropped frame dimensions (visual presentation size) of the AVC stream that is described by that entry. The width and height fields do *not* reflect any changes in size caused by SEI messages such as pan-scan. The visual handling of SEI messages such as pan-scan is both optional and terminal-dependent. If the width and height of the sequence changes, then a new sample description is needed.

Note that the visual size in the SPS may be either frame or field size; in the sample entry, it is always the frame size.

The width and height fields in the track header may not be the same as the width and height fields in the one or more `VisualSampleEntry` in the video track. As specified in the ISO Base Media File Format, if normalized visual presentation is needed, all the sequences are normalized to the track width and height for presentation.

5.3.8 Parameter sets

This subclause applies when a separate parameter set stream is not used.

Each AVC sample description, which contains the AVC video stream decoder specific information, includes a group of SPSs and PPSs. This group of parameter sets functions much like a codebook. Each parameter set has an identifier, and each slice references the parameter set it was coded against using the parameter set's identifier.

In the file format each configuration of parameter sets is represented separately. A parameter set cannot be updated without causing a different sample description to be used. For example, suppose that samples 1 to 4 use PPSs identified as 1, 2, 3 and a single SPS identified as 1. At sample 5 a new value of PPS 2 is required but PPSs 1 and 3 remain unaltered and are used until sample 10. In this case, the sample description for samples 1 through 4 is the same and contains the initial values of PPSs 1, 2, 3 and SPS 1. At sample 5 the sample description must change to a second sample description, which contains the updated value for PPS 2 *as well as* the original values of PPSs 1 and 3 and SPS 1. This second sample description is used for samples 5 through 10.

Systems wishing to send SPS or PPS updates will need to compare the two configurations to find the differences in order to send the appropriate parameter set updates.

NOTE 1 It is recommended that when several parameter sets are used and parameter set updating is desired, a separate parameter set elementary stream be used.

NOTE 2 Decoders conforming to this specification are required to support both parameter sets stored in separate elementary streams as well as parameter sets stored in the AVC sample description entries, unless restricted by another specification using this one.

5.3.9 Decoding time (DTS) and composition time (CTS)

Samples are stored in the file format in decoding order. If picture reordering is not used and decoding and composition times are the same, then presentation is the same as decoding order and only the time-to-sample ‘stts’ table is used. Note that any kind of picture may be reordered in AVC video, not only B-pictures.

If decoding time and composition time differ, the composition time-to-sample ‘ctts’ table is also used in conjunction with the 'stts' table.

5.3.10 Sync sample (IDR)

An AVC sample is considered as a random access point if ALL of the following conditions are met:

- The video data NAL units in the sample indicate that the primary picture contained in the sample is an instantaneous decoding refresh (IDR) picture.

- All SPSs and PPSs needed to decode the video data NAL units in the sample of the IDR picture are contained in the decoder configuration of the video elementary stream or in a separate parameter set elementary stream sample.

A parameter set elementary stream sample is a random access point if and only if all parameter sets required by the associated video elementary stream from the time of the parameter set sample forward are supplied, in the parameter set stream, before they are required by the associated video elementary stream.

5.3.11 Shadow sync

The use of the shadow sync table to indicate alternate encodings of a sample for random access are supported as defined in the ISO Base Media File Format. A shadow sync shall indicate a sample that is a random access point as specified in 5.3.10.

While the use of shadow sync for AVC content is supported for backward compatibility reasons, this use is deprecated and use of the mechanisms defined in 5.3.13 is recommended.

5.3.12 Layering and sub-sequences

5.3.12.1 Introduction

Streams may be constructed so that the referential dependencies between samples allow only subsets of the samples to be sent to the decoder. This mechanism is called *thinning* a stream. Thinning discards entire sets of samples using knowledge of what other sets of pictures this set of pictures depends on and what picture sets in turn depend on it.

The referential dependencies between samples in a stream are structured into *layers* and *sub-sequences*. Samples in higher layers can only depend on samples in lower layers. Layers are numbered, and the samples are organized such that a sample in layer N has no dependencies on samples in layers greater than N.

Sub-sequences are as defined in the Annex D of ISO/IEC 14496-10. Dependency relations between sub-sequences represent the dependency structure of a stream. Each sub-sequence belongs to one and only one layer. A sample shall reside in one layer and in one sub-sequence only.

Layering and sub-sequence information is represented in the file format to allow systems reading the files to understand the ways in which stream thinning may be achieved without having to examine the dependency structure of every sample.

Layer and sub-sequences are represented in the AVC file format as Sample Group. An AVC file shall contain zero or one instance of a `SampleGroupBox` (per track) with a `grouping_type` equal to 'layr'. This `SampleGroupBox` instance represents the assignment of samples in a track to layers. An accompanying instance of the SampleGroupDescriptionBox with the same grouping type shall, if it exists, contain `AVCLayerEntry` sample group entries describing the layers. Similarly, an AVC file shall contain zero or one instance of a `SampleGroupBox` (per track) with a `grouping_type` equal to 'sseq'. This `SampleGroupBox` instance represents the assignment of samples in a track to sub-sequences. An accompanying instance of the SampleGroupDescriptionBox with the same grouping type shall, if it exists, contain `AVCSubSequenceEntry` sample group entries describing the sub-sequences.

5.3.12.2 Sub-sequence description entry

Box Types: ‘avss'
Container: Sample Group Description Box ('sgpd')
Mandatory: No
Quantity: Zero or more.

A sub-sequence description entry is a sample group entry that describes a sub-sequence. A sub-sequence is a set of samples in a track belonging to the same layer. A sub-sequence depends on another sub-sequence if and only if there exists a sample in the sub-sequence that is directly referentially dependent on some sample in the other sub-sequence. All samples in a sub-sequence shall directly depend only on (i.e., refer to) other samples within the same sub-sequence or samples in the sub-sequences on which is it dependent. A sub-sequence can depend on zero or more sub-sequences in the lower layers. A sub-sequence shall not depend on any other sub-sequence in the same or higher layer.

At most one partition of an AVC stream into layers shall exist in the AVC file format; that is, there is either zero or one instances of the sample group boxes (`SampleGroupBox`, `SampleGroupDescriptionBox`) per track concerning the grouping of samples into layers and sub-sequences.

5.3.12.2.1 Syntax

```
aligned(8) class DependencyInfo
{
      unsigned int(8)   subSeqDirectionFlag;
      unsigned int(8)   layerNumber;
      unsigned int(16)  subSequenceIdentifier;
}
```

```
class AVCSubSequenceEntry () extends VisualSampleGroupEntry ('avss')
{
      unsigned int(16) subSequenceIdentifer;
      unsigned int(8)  layerNumber;
      unsigned int(1)  durationFlag;
      unsigned int(1)  avgRateFlag;
      unsigned int(6)  reserved = 0;
      if (durationFlag)
         unsigned int(32) duration;
      if (avgRateFlag)
      {
         unsigned int(8)  accurateStatisticsFlag;
         unsigned int(16) avgBitRate;
         unsigned int(16) avgFrameRate;
      }
      unsigned int(8) numReferences;
      DependencyInfo dependency[numReferences];
   }
}
```

5.3.12.2.2 Semantics

`subSeqDirectionFlag`, `layerNumber` and `subSequenceIdentifier` within the `DependencyInfo` class identify a sub-sequence that is used as a reference for this sub-sequence. Only direct, not indirect, referential dependencies shall be identified. The identified sub-sequence has sub-sequence identifier equal to `subSequenceIdentifier` and resides in the layer having the layer number equal to `layerNumber`. If `subSeqDirectionFlag` is 0, the sub-sequence used as a reference for this sub-sequence is the closest sub-sequence among all the candidate sub-sequences whose first picture precedes the first picture of this sub-sequence in decoding order and which resides in the indicated layer and has the indicated sub-sequence identifier; 'closest' means that among all the candidate sub-sequences the first picture of the referenced sub-sequence is the closest to the first picture of this sub-sequence in decoding order. If `subSeqDirectionFlag` is equal to 1, the sub-sequence used as a reference for this sub-sequence is the closest sub-sequence among all the candidate sub-sequences whose first picture succeeds the first picture of this sub-sequence in decoding order and which resides in the indicated layer and has the indicated sub-sequence identifier; 'closest' has the same meaning as above.

`subSequenceIdentifier` gives the identifier for the sub-sequence.

`layerNumber` gives the layer number to which the sub-sequence belongs.

`durationFlag` equal to 0 indicates that the duration of the target sub-sequence is not specified. Otherwise, a value of 1 indicates that the `duration` field indicates the duration of this sub-sequence.

`avgRateFlag` equal to 0 indicates that the average bit rate and the average frame rate of the target sub-sequence are unspecified. Otherwise, a value of 1 indicates that the average rate characteristics are described by the `accurateStatisticsFlag`, `avgBitRate`, and `avgFrameRate` fields.

`duration` indicates the duration of the target sub-sequence in clock ticks of a 90-kHz clock.

`accurateStatisticsFlag` indicates how reliable the values of `avgBitRate` and `avgFrameRate` are. `accurateStatisticsFlag` equal to 1 indicates that `avgBitRate` and `avgFrameRate` are rounded from statistically correct values. `accurateStatisticsFlag` equal to 0 indicates that `avgBitRate` and `avgFrameRate` are estimates and may deviate somewhat from the correct values.

`avgBitRate` gives the average bit rate in (1000 bits)/second of this sub-sequence. All NAL units of this sub-sequence are taken into account in the calculation. In the following, B is the number of bits in all NAL units in the sub-sequence. t_1 is the decoding timestamp of the first picture of the sub-sequence (in decoding order), and t_2 is the decoding timestamp of the last picture of the sub-sequence (in decoding order). Then, the avgBitRate is calculated as follows provided that $t_1 \neq t_2$: avgBitRate = round($B \div ((t_2 - t_1) * 1000)$). If $t_1 = t_2$, avgBitRate shall be 0.

`avgFrameRate` gives the average frame rate in units of frames/(256 seconds) of this sub-sequence. All NAL units of this sub-sequence are taken into account in the calculation. The average frame rate is

calculated according to the presentation timestamp of the frame. In the following, C is the number of frames in the sub-sequence. t_1 is the presentation timestamp of the first picture of the sub-sequence (in decoding order), and t_2 is the presentation timestamp (in seconds) of the last picture of the sub-sequence (in decoding order). Then, the avgFrameRate is calculated as follows provided that $t_1 \neq t_2$: avgFrameRate = round(C * 256 ÷ (t_2 – t_1)). If $t_1 = t_2$, `avgFrameRate` shall be 0. Value zero indicates an unspecified frame rate.

`numReferences` gives the number of sub-sequences directly referenced in this sub-sequence. `dependency` is an array of `DependencyInfo` structures giving the identifying referenced sub-sequences.

5.3.12.3 Layer description entry

Box Types: 'avll'
Container: Sample Group Description Box ('sgpd')
Mandatory: No
Quantity: Zero or more.

A layer sample group entry defines the layer information for all samples in a layer. Layers are numbered with non-negative integers. Layers are ordered hierarchically based on their dependency on each other: A layer having a larger layer number is a higher layer than a layer having a smaller layer number. The layers are ordered hierarchically based on their dependency on each other so that a layer does not depend on any higher layer and may depend on lower layers. The lowest layer is numbered as zero and other layers are given consecutive numbers. In other words, layer 0 is independently decodable, pictures in layer 1 may be predicted from layer 0, pictures in layer 2 may be predicted from layers 0 and 1, etc.

5.3.12.3.1 Syntax

```
class AVCLayerEntry() extends VisualSampleGroupEntry ('avll')
{
   unsigned int(8)  layerNumber;
   unsigned int(8)  accurateStatisticsFlag;
   unsigned int(16) avgBitRate;
   unsigned int(16) avgFrameRate;
}
```

5.3.12.3.2 Semantics

`layerNumber` gives the number of this layer with the base layer being numbered as zero and all enhancement layers being numbered as one or higher with consecutive numbers.

`accurateStatisticsFlag` indicates how reliable the values of `avgBitRate` and `avgFrameRate` are. `accurateStatisticsFlag` equal to 1 indicates that `avgBitRate` and `avgFrameRate` are rounded from statistically correct values. `accurateStatisticsFlag` equal to 0 indicates that `avgBitRate` and `avgFrameRate` are estimates and may deviate somewhat from the correct values.

`avgBitRate` gives the average bit rate in units of 1000 bits per second. All NAL units in this and lower sub-sequence layers are taken into account in the calculation. The average bit rate is calculated according to the decoding timestamp. In the following, B is the number of bits in all NAL units in this and lower sub-sequence layers. t_1 is the decoding timestamp of the first picture in this and lower sub-sequence layers in the presentation order, and t_2 is the decoding timestamp of the latest picture in this and lower sub-sequence layers in the presentation order. Then, `avgBitRate` is calculated as follows provided that $t_1 \neq t_2$: `avgBitRate` = round(B ÷ ((t_2 – t_1) * 1000))). If $t_1 = t_2$, `avgBitRate` shall be 0. Value zero indicates an unspecified bit rate.

`avgFrameRate` gives the average frame rate in units of frames/(256 seconds). All NAL units in this and lower sub-sequence layers are taken into account in the calculation. In the following, C is the number of frames in this and lower sub-sequence layers. t_1 is the presentation timestamp of the first picture in this and lower sub-sequence layers in presentation order, and t_2 is the presentation timestamp of the latest picture in this and lower sub-sequence layers in the presentation order. Then, the

`avgFrameRate` is calculated as follows provided that $t_1 \neq t_2$: avgFrameRate = round(C * 256 ÷ (t_2 – t_1)). If $t_1 = t_2$, `avgFrameRate` shall be 0. Value zero indicates an unspecified frame rate.

5.3.13 Alternate streams and switching pictures

In typical streaming scenarios, one of the key requirements is to scale the bit rate of the compressed data in response to changing network conditions. The simplest way to achieve this is to encode multiple streams with different bandwidths and quality settings for representative network conditions. The server can then switch amongst these pre-coded streams in response to network conditions. In earlier standards, switching between streams is only possible at I-pictures, because the pictures can only be switched when there are no dependencies on prior pictures for reconstruction.

AVC includes supports for SP-pictures and SI-pictures ("switching pictures") that allow switching from one stream to another while still supporting inter coding of switching pictures. Figure 3 shows how SP pictures are used to switch between two different bit streams.

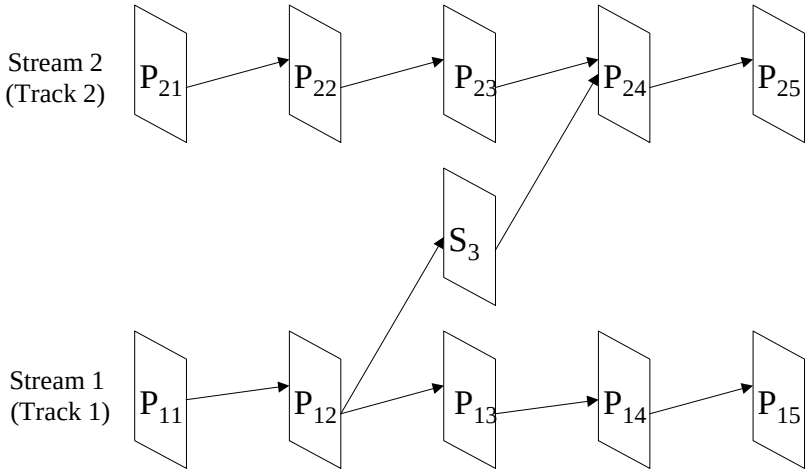


Figure 3 — Stream switching

In the file format, switching pictures are stored in *switching picture* tracks, which are tracks separate from the track that is being switched from and the track being switched to. Switching picture tracks can be identified by the existence of a specific required track reference in that track. A switching picture is an alternative to the sample in the destination track that has exactly the same decoding time. If all switching pictures are SI pictures, then no further information is needed.

If any of the pictures in the switching track are SP pictures, then two extra pieces of information may be needed. First, the source track that is being switched from must be identified by using a track reference (the source track may be the same track as the destination track). Second, the dependency of the switching picture on the samples in the source track may be needed, so that a switching picture is only used when the pictures on which it depends have been supplied to the decoder.

This dependency is represented by means of an optional extra sample table. There is one entry per sample in the switching track. Each entry records the relative sample numbers in the source track on which the switching picture depends. If this array is empty for a given sample, then that switching sample contains an SI picture. If the dependency box is not present, then only SI-frames shall be present in the track.

A switching sample may have multiple coded representations with different dependencies. For AVC video, the multiple representations of a switching sample are stored in different switching tracks (i.e. access unit). For example, one switch track might contain a SP-picture representation dependent on some earlier samples, used for stream switching, while another switch track may contain another representation as an SI-picture, used for random access.

5.3.13.1 Alternate group

The ISO Base Media File Format (but not the version one specification of the MPEG-4 file format, which is branded as 'mp41') supports the use of what is called *alternate tracks.* Each track can optionally specify an *alternate group* (in the track header box) that groups together alternate encodings of the same content. Thus, each alternate bit-stream can be stored as a separate track and related together as alternate tracks. All the tracks which form a group which may be switched between, but not the tracks containing the switching pictures, must be a member of an `alternate_group` with a non-zero group identifier.

An alternate group is not needed if there is only one primary track, with a switching track. This switching track may contain SI pictures, or SP pictures for trick modes or error resilience, which predict both from and to the same track.

5.3.13.2 Track references

The switching track must be linked to the track into which it switches (the destination track) by a track reference of type ‘swto’ in the switching picture track.

If the switching track contains SP pictures, the switching track must be linked to the track from which it switches (the source track) by a track reference of type ‘swfr’ in the switching picture track.

5.3.13.3 Sample dependency

This subclause defines the dependencies of each switching sample on sample(s) in the source track. This table is only needed in a switching track that has a source (‘swfr’) track dependency.

5.3.13.3.1 Definition

Box Type: ‘sdep’
Container: Sample Table ‘stbl’
Mandatory: No
Quantity: Zero or exactly one.

This box contains the sample dependencies for each switching sample. The dependencies are stored in the table, one record for each sample. The size of the table, `sample_count` is taken from the `sample_count` in the Sample Size Box ('stsz') or Compact Sample Size Box (‘stz2’).

5.3.13.3.1.1 Syntax

```
aligned(8) class SampleDependencyBox
   extends FullBox('sdep', version = 0, 0) {
   for (i=0; i < sample_count; i++){
      unsigned int(16) dependency_count;
      for (k=0; k < dependency_count; k++) {
         signed int(16) relative_sample_number;
      }
   }
}
```

5.3.13.3.1.2 Semantics

`dependency_count` is an integer that counts the number of samples in the source track on which this switching sample directly depends.

`relative_sample_number` is an integer that identifies a sample in the source track. The relative sample numbers are encoded as follows. If there is a sample in the source track with the same decoding time, it has a relative sample number of 0. Whether or not this sample exists, the sample in the source track which immediately precedes the decoding time of the switching sample has relative sample number –1, the sample before that –2, and so on. Similarly, the sample in the source track

which immediately follows the decoding time of the switching sample has relative sample number +1, the sample after that +2, and so on.

5.3.14 Random access recovery points

The AVC codec includes the concept of a 'gradual decoding refresh' or random access recovery point. This is signalled in the bit-stream using the recovery point SEI message. This message is found at the beginning of the random access, and indicates how much data must be decoded subsequent to the access unit at the position of the SEI message before the recovery is complete.

This concept of gradual recovery is supported in the file format also by using RollRecoveryEntry Groups [4.5]. In order that the group membership marks the sample containing the SEI message the `'roll-distance'` is constrained to being only positive (i.e. a post-roll).

Note carefully that the roll-group counts samples in the file format; this may not match the way that the distances are represented in the SEI message.

Within a stream, it is necessary to mark the beginning of the pre-roll, so that a stream decoder may start decoding there. However, in a file, when performing random access, a deterministic search is desired for the closest preceding frame which can be decoded perfectly (either a sync sample, or the end of a pre-roll).

5.3.15 Hinting

Note that what the hint tracks call "B frames" are actually 'disposable' pictures or non-reference pictures as defined in ISO/IEC 14496-10.