INTERNATIONAL STANDARD

ISO/IEC 14496-15

Fifth edition
2019-09

Information technology — Coding of audio-visual objects —

Part 15:
Carriage of network abstraction layer (NAL) unit structured video in the ISO base media file format

Technologies de l'information — Codage des objets audiovisuels —

Partie 15: Transport de vidéo structurée en unités NAL sur la couche réseau au format ISO de base pour les fichiers médias

Contents

Page

iii

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

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

This fifth edition cancels and replaces the fourth edition (ISO/IEC 14496-15:2017), which has been technically revised. It also incorporates Amendments ISO/IEC 14496-15:2017/Amd.1:2018 and ISO/IEC 14496-15:2017/Amd.2:2019.

The main changes compared to the previous edition are as follows:

— additional content incorporated as subclauses 4.11, 4.12, 9.6.4, D.4.3, D.4.4 and D.4.5 and Annex F;

— corrections in Tables 2, 3 and 6 and subclause A.1;

— deletion of subclause 5.4.10;

— minor editorial changes to align the document with the drafting rules in ISO/IEC Directives Part 2.

A list of all parts in the ISO/IEC 14496 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

This document defines a storage format based on, and compatible with, the ISO base media file format (ISO/IEC 14496-12), 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 document enables video streams formatted as network adaptation layer units (NAL units) to

a)   be used in conjunction with other media streams, such as audio,

b)   be used in an MPEG-4 systems environment, if desired,

c)   be formatted for delivery by a streaming server, using hint tracks, and

d)   inherit all the use cases and features of the ISO base media file format on which MP4 and MJ2 are based.

This document can be used as a standalone specification; it specifies how NAL unit structured video content is 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 NAL unit structured video such as AVC (ISO/IEC 14496-10) video and high efficiency video coding (HEVC, ISO/IEC 23008-2) 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.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights. The holders of these patent rights have assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC. Informstion may be obtained from:

Fraunhofer-Gesellschaft
Hansastr. 27c
80686 München
Germany

Nokia Corporation
PO Box 86
FIN-24101 Salo
Finland

TDVision Systems, Inc.
8001 Irvine Center Drive, Suite 400
Irvine, CA 92618
USA

Qualcomm Incorporated
5775 Morehouse Drive,
San Diego, CA 92121
USA

Huawei Technologies Co., Ltd.
Bantian Longgang District,
Shenzhen 518129
China

Telefonaktiebolaget LM Ericsson (publ)
Ericsson AB,
SE-164 80 Stockholm
Sweden

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

Information technology — Coding of audio-visual objects —

Part 15:
Carriage of network abstraction layer (NAL) unit structured video in the ISO base media file format

1 Scope

This document specifies the storage format for streams of video that is structured as NAL units, such as AVC (ISO/IEC 14496-10) and HEVC (ISO/IEC 23008-2) video streams.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements 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 14496-10:2014, *Information technology — Coding of audio-visual objects — Part 10: Advanced Video Coding*

ISO/IEC 14496-12:2015, *Information technology — Coding of audio-visual objects — Part 12: ISO base media file format*

ISO/IEC 23008-2:2017, *Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding*

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 14496-10, ISO/IEC 23008-2 and the following apply.

3.1.1
3D-AVC NAL unit
3D-AVC VCL NAL unit
NAL unit with type 21 with avc_3d_extension_flag equal to 1

Note 1 to entry: As specified in ISO/IEC 14496-10:2014, Annex J.

3.1.2
aggregator
in-stream structure (3.1.11) using a NAL unit header for grouping of NAL units belonging to the same sample

3.1.3
AVC base layer
maximum subset of a bitstream that is AVC compatible

Note 1 to entry: This can also be expressed as a bitstream not using any of the functionality of ISO/IEC 14496-10:2014 Annex G, Annex H, Annex I, or Annex J.

Note 2 to entry: The AVC base layer is represented by AVC VCL NAL units and associated non-VCL NAL units.

Note 3 to entry: The AVC base layer itself can be a temporal scalable bitstream.

3.1.4
AVC NAL unit
AVC VCL NAL unit (3.1.5) and its associated non-VCL NAL units in a bitstream

3.1.5
AVC VCL NAL unit
NAL unit with type 1 to 5 (inclusive) Note 1 to entry: As specified in ISO/IEC 14496-10.

3.1.6
complete subset
minimal set of tracks that contain all the information in the original bitstream

3.1.7
cropped frame dimensions
width and height of the decoded frame after applying the output cropping parameters specified by the active SPS

3.1.8
extraction path
set of operations on the original bitstream, each yielding a subset bitstream, ordered such that the complete bitstream is first in the set, and the base layer is last, and all the bitstreams are in decreasing complexity (along one of the scalability axes, such as resolution) and where every bitstream is a valid operating point

Note 1 to entry: An extraction path can be represented by the values of priority_id in the NAL unit headers. Alternatively, an extraction path can be represented by the run of tiers or by a set of hierarchically dependent tracks.

3.1.9
extractor
in-stream structure (3.1.11) using a NAL unit header for extraction of data from other tracks

Note 1 to entry: Extractors contain instructions on how to extract data from other tracks. Logically an Extractor can be seen as a pointer to data. While reading a track containing Extractors, the Extractor is replaced by the data it is pointing to.

3.1.10
implicit reconstruction
reconstruction of a stream of access units from two or more tracks not using extractors

3.1.11
in-stream structure
structure residing within sample data

3.1.12
layer set
set of layers represented within a bitstream created from another bitstream by operation of the sub-bitstream extraction process

Note 1 to entry: As specified in ISO/IEC 23008-2.

3.1.13
MVC NAL unit
MVC VCL NAL unit (3.1.14) and its associated non-VCL NAL units in an MVC streamNote 1 to entry: As specified in ISO/IEC 14496-10:2014, Annex H.

3.1.14
MVC VCL NAL unit
NAL unit with type 20, and NAL units with type 14 when the immediately following NAL units are AVC VCL NAL units

Note 1 to entry: As specified in ISO/IEC 14496-10.

Note 2 to entry: MVC VCL NAL units do not affect the decoding process of a legacy AVC decoder.

3.1.15
MVC+D depth NAL unit
MVC+D depth VCL NAL unit
NAL unit with type 21 containing a coded slice extension for a depth view component Note 1 to entry: As specified in ISO/IEC 14496-10:2014 Annex I.

3.1.16
MVD NAL unit
MVD VCL NAL unit
NAL unit with type 21, containing a coded slice extension for a depth view component coded with MVC+D or 3D-AVC, or a 3D-AVC texture view component

Note 1 to entry: As specified in ISO/IEC 14496-10:2014, Annex I or J.

3.1.17
NAL-unit-like structure
data structure that is similar to NAL units in the sense that it also has a NAL unit header and a payload, with a difference that the payload might not follow the start code emulation prevention mechanism required for the NAL unit syntax

Note 1 to entry: As specified in ISO/IEC 14496-10 or ISO/IEC 23008-2.

3.1.18
natively present
not included in an *aggregator* ([3.1.2](#)) or an *extractor* ([3.1.9](#))

Note 1 to entry: Data referred to by (hence not included in) an aggregator is considered as natively present. Data included in an aggregator is not considered as natively present.

3.1.19
operating point
<AVC and SVC elementary stream> independently decodable subset of a layered bitstream

Note 1 to entry: Each operating point consists of all the data needed to decode this particular bitstream subset.

Note 2 to entry: In an SVC stream an operating point represents a particular spatial resolution, temporal resolution and quality, and can be represented either by (i) specific values of DTQ (dependency_id, temporal_id and quality_id) or (ii) specific values of P (priority_id) or (iii) combinations of them (e.g. PDTQ). Note that the usage of priority_id is defined by the application. In an SVC file a track represents one or more operating points. Within a track tiers can be used to define multiple operating points.

Note 3 to entry: The bitstream subset of an MVC or MVD operating point represents a particular set of target output views at a particular temporal resolution and consists of all the data needed to decode this particular bitstream subset. In MVD each target output view in the bitstream subset of an MVD operating point can contain a texture view, a depth view or both.

Note 4 to entry: An operating point is referred to as an operation point in ISO/IEC 14496-10:2014, Annex H or an output operation point in ISO/IEC 23008-2.

3.1.20
operating point
<HEVC, layered HEVC and tiled HEVC elementary stream> independently decodable subset of a layered bitstream, where one or more layers in the set of layers are indicated to be output layers

Note 1 to entry: Each operating point consists of all the data needed to decode this particular bitstream subset.

Note 2 to entry: In an SVC stream an operating point represents a particular spatial resolution, temporal resolution, and quality, and can be represented either by (i) specific values of DTQ (dependency_id, temporal_id and quality_id) or (ii) specific values of P (priority_id) or (iii) combinations of them (e.g. PDTQ). Note that the usage of priority_id is defined by the application. In an SVC file a track represents one or more operating points. Within a track tiers can be used to define multiple operating points.

Note 3 to entry: The bitstream subset of an MVC or MVD operating point represents a particular set of target output views at a particular temporal resolution and consists of all the data needed to decode this particular bitstream subset. In MVD each target output view in the bitstream subset of an MVD operating point can contain a texture view, a depth view or both.

Note 4 to entry: An operating point is referred to as an operation point in ISO/IEC 14496-10:2014, Annex H or an output operation point in ISO/IEC 23008-2.

3.1.21
output layer set
set of layers consisting of the layers of one of the specified *layer sets* (3.1.12), where one or more layers in the set of layers are indicated to be output layers

Note 1 to entry: As specified in ISO/IEC 23008-2.

3.1.22
parameter set
video parameter set, sequence parameter set or picture parameter set

Note 1 to entry: As defined in the applicable video standard (e.g. ISO/IEC 14496-10 or ISO/IEC 23008-2).

Note 2 to entry: This term is used to refer to all types of parameter sets.

3.1.23
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* (3.1.40)

3.1.24
picture unit
set of VCL NAL units and their associated non-VCL NAL units

Note 1 to entry: As specified in ISO/IEC 23008-2.

3.1.25
prefix NAL unit
NAL units with type 14

Note 1 to entry: As specified in ISO/IEC 14496-10.

Note 2 to entry: Prefix NAL units provide scalability information about AVC VCL NAL units and filler data NAL units. Prefix NAL units do not affect the decoding process of a legacy AVC decoder. The behaviour of a legacy AVC file reader as a response to prefix NAL units is undefined.

3.1.26
reference layer
layer that is indicated as possibly needed for decoding of another layer

Note 1 to entry: As specified in ISO/IEC 23008-2 and as specified by the `'oinf'` sample group defined in subclause 9.6.2.

3.1.27
scalable layer
layer
<AVC and SVC elementary stream> set of VCL NAL units with the same values of dependency_id, quality_id, and temporal_id, and the associated non-VCL NAL units

Note 1 to entry: As specified in ISO/IEC 14496-10.

Note 2 to entry: A scalable layer with any of dependency_id, quality_id, and temporal_id not equal to 0 enhances the video by one or more scalability levels in at least one direction (temporal, quality or spatial resolution).

Note 3 to entry: SVC uses a "layered" encoder design that results in a bitstream representing "coding layers". In some publications the 'base layer' is the first quality layer of a specific coding layer. In some publications the base layer is the scalable layer with the lowest priority. The SVC file format uses "scalable layer" or "layer" in a general way for describing nested bitstreams (using terms like AVC base layer or SVC enhancement layer).

3.1.28
scalable layer
layer
<HEVC, layered HEVC and tiled HEVC elementary stream> set of VCL NAL units with the same value of nuh_layer_id and the associated non-VCL NAL units

Note 1 to entry: As specified in ISO/IEC 23008-2.

3.1.29
scalable layer representation
bitstream subset that is required for decoding the *scalable layer* (3.1.27/3.1.28), consisting of the scalable layer itself and all the scalable layers on which the scalable layer depends

Note 1 to entry: A scalable layer representation is also referred to as the representation of the scalable layer.

3.1.30
sub-picture
proper subset of coded slices of a layer representation

3.1.31
sub-picture tier
tier that consists of *sub-pictures* (3.1.30)

Note 1 to entry: Any coded slice that is not included in the tier representation of a sub-picture tier is not to be referred to in inter prediction or inter-layer prediction for decoding of the sub-picture tier.

3.1.32
sub-layer
set of VCL NAL units with a particular value of TemporalId and the associated non-VCL NAL units

Note 1 to entry: As specified in ISO/IEC 23008-2.

3.1.33
SVC enhancement layer
layer that specifies a part of a scalable bitstream that enhances the video

Note 1 to entry: An SVC enhancement layer is represented by SVC VCL NAL units and the associated non-VCL NAL units and SEI messages.

Note 2 to entry: Usually an SVC enhancement layer represents a spatial or coarse-grain scalability (CGS) coding layer (identified by a specific value of dependency_id).

3.1.34
SVC NAL unit
SVC VCL NAL unit (3.1.36) and its associated non-VCL NAL units in an *SVC stream* (3.1.35)

Note 1 to entry: As specified in ISO/IEC 14496-10:2014, Annex G.

3.1.35
SVC stream
bitstream represented by the operating point for which dependency_id is equal to mDid, temporal_id is the greatest temporal_id value among mOpSet, and quality_id is the greatest quality_id value among mOpSet, where the greatest value of dependency_id of all the operating points represented by DTQ (dependency_id, temporal_id and quality_id) combinations is equal to mDid, and the set of all the operating points with dependency_id equal to mDid is mOpSet

Note 1 to entry: The term "SVC stream" is referenced by 'decoding/accessing the entire stream' in this document. There can be NAL units that are not required for decoding this operating point.

3.1.36
SVC VCL NAL unit
NAL unit with type 20, and NAL units with type 14 when the immediately following NAL units are AVC VCL NAL units

Note 1 to entry: As specified in ISO/IEC 14496-10:2014, Annex G.

Note 2 to entry: SVC VCL NAL units do not affect the decoding process of a legacy AVC decoder.

3.1.37
temporal layer representation
representation of a temporal layer
temporal layer and all lower temporal layers

3.1.38
tier
set of operating points within a track, providing information about the operating points and instructions on how to access the corresponding bitstream portions (using maps and groups)

Note 1 to entry: In SVC file format a tier represents one or more scalable layers of an SVC bitstream. In the context of ISO/IEC 23008-2 video, the term tier is used to represent a part of the interoperability point representation consisting of profile, tier, and level. Readers should not be confused about these two different meanings of the word "tier".

Note 2 to entry: The term "tier" is used in SVC file format to avoid confusion with the frequently used term layer. A tier represents a subset of a track and represents an operating point of an SVC bitstream. Tiers in a track subset the entire track, no matter whether the track references another track by extractors.

Note 3 to entry: An MVC or MVD tier represents a particular set of temporal subsets of a particular set of views.

3.1.39
tier representation
representation of the tier
bitstream subset that is required for decoding the tier, consisting of the tier itself and all the tiers on which the tier depends

3.1.40
video elementary stream
elementary stream containing access units made up of NAL units for coded picture data

3.1.41
video stream
self-contained independently decodable video bitstream

3.1.42
virtual base view
AVC compatible representation of an independently coded non-base view

Note 1 to entry: As specified in ISO/IEC 14496-10:2014, Annex H.

Note 2 to entry: The virtual base view of an independently coded non-base view is created according to the process specified in ISO/IEC 14496-10:2014, H.8.5.5. Samples containing data units of an independently coded non-base view and samples of the virtual base view are aligned by decoding times.

3.2 Abbreviated terms

3D-AVC   three-dimensional advanced video coding
(refers to ISO/IEC 14496-10:2014, Annex J)

3D-HEVC   three-dimensional high efficiency video coding
(refers to ISO/IEC 23008-2:2017, Annex I)

A3D   three-dimensional advanced video coding
(refers to ISO/IEC 14496-10:2014, Annex G)

NOTE 1   The abbreviation A3D is used in terminology related to syntax elements and structures, whereas the abbreviation 3D-AVC is used otherwise.

AVC   advanced video coding
[where contrasted with SVC, MVC, or MVD in this document, this term refers to ISO/IEC 14496-10, excluding its Annex G (Scalable Video Coding), Annex H (Multiview Video Coding), Annex I (Multiview and Depth Video Coding), and Annex J (Multiview and Depth Video with Enhanced Non-Base View Coding)]

BLA   broken link access

CRA   clean random access

CTU   coding tree unit

HEVC   high efficiency video coding

FF   file format

HRD   hypothetical reference decoder

IDR   instantaneous decoding refresh

IRAP   intra random access point

L-HEVC   layered high efficiency video coding

MVC   multiview video coding
(refers to ISO/IEC 14496-10:2014, Annex H)

MVCD   multiview video coding plus depth
(refers to ISO/IEC 14496-10:2014, Annex I)

MVC+D   multiview video coding plus depth
(refers to ISO/IEC 14496-10:2014, Annex I)

NOTE 2   The abbreviation MVCD is used in terminology related to syntax elements and structures, whereas the abbreviation MVC+D is used otherwise.

MV-HEVC   multiview high efficiency video coding
(refers to ISO/IEC 23008-2:2017, Annex G)

MVD   multiview video coding plus depth
(refers to ISO/IEC 14496-10:2014, Annexes I and J)

NAL	network abstraction layer
PPS	picture parameter set
RBSP	raw byte sequence payload
ROI	region-of-interest
SAP	stream access point
SEI	supplementary enhancement information
SHVC	scalable high efficiency video coding (refers to ISO/IEC 23008-2:2017, Annex H)
SPS	sequence parameter set
STSA	step-wise temporal sub-layer access
SVC	scalable video coding (refers to ISO/IEC 14496-10:2014, Annex G)
TSA	temporal sub-layer access
VCL	video coding layer
VPS	video parameter set

4   General definitions

4.1   Overview

The specifications in this clause apply to all coding systems identified by this document, unless specifically over-ridden by definitions in the clause for a specific coding system.

Table 1 summarizes the correspondences between the sets of terminology used in video specifications and the ISO base media file format for the AVC file format specified in subclause 4.11 and the HEVC file format specified in Clause 8 (it does not appy for all cases for Clauses 6, 7 and 9).

Table 1 — Correspondence of terms in video and ISO base media file format

Video	ISO base media file format
—	movie
bitstream	track
access unit	sample

4.2   Elementary stream structure

This document concerns video coding systems that specify 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 content.

4.3   Sample and configuration definition

4.3.1   General

Sample: A sample is an access unit or a part of an access unit (e.g., in a track containing a part of a multi-layer video bitstream), where an access unit is as defined in the appropriate specification.

Parameter set sample: A parameter set sample is a sample in a parameter set stream that 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.

4.3.2   Canonical order and restrictions

The 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.

When multiple tracks are used to store an elementary stream, as may be the case for Clauses 6, 7 and 9, some tracks may contain canonical streams while others may need to be processed (e.g., when extractors are used or when an implicit reconstruction of access units is needed) to produce a canonical stream.

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

— **Video data NAL units**: In the context of subclause 4.11 or Clause 8, all video data NAL units for a single picture shall be contained within the sample whose decoding time and composition time are those of the picture.

— **SEI NAL units**: All SEI NAL units shall be contained in the parameter set arrays, or in the sample whose decoding time is at the time, or immediately precedes the time (with no intervening samples), when the SEI messages come into effect instantaneously. In general, 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. The order of SEI messages within a sample is as defined in the applicable video coding standard.

— **NAL unit order**: 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 the applicable video coding standard.

— **All timing information is external to stream:** Picture Timing SEI messages that define presentation or composition timestamps may be included in the video elementary stream, as these messages contain 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 overrides any timing provided in the video layer. Timing provided within the video 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 can invalidate or make inconsistent any embedded timing information present within the video stream.

— **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 subclause 4.3.3; 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.

4.3.3 Sample format

4.3.3.1 Definition

This subclause defines the structure of the samples. Samples are externally framed and have a size supplied by that external framing. The syntax of a sample is configured via the decoder specific configuration for the elementary stream. An example of the structure of a video sample is depicted in the following figure.
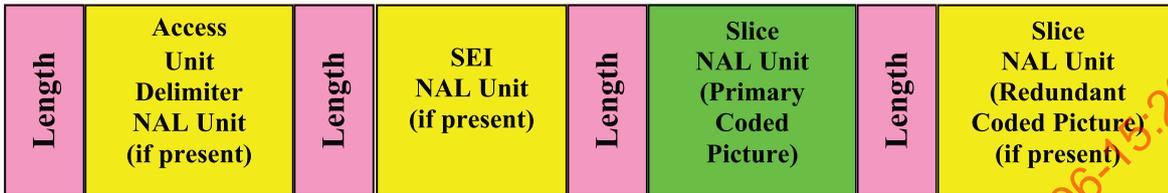
Figure 1 — Example structure of a sample

An 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 may be configured to be of 1, 2, or 4 bytes.

— *NAL Unit*: Contains the NAL unit data as specified in the applicable video coding standard.

4.3.3.2 Syntax

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

4.3.3.3 Semantics

DecoderConfigurationRecord indicates the record in the matching sample entry (e.g. AVCDecoderConfigurationRecord in the case of AVC).

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

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

4.3.4 Optional boxes in the sample entry

Unless otherwise explicitly specified, the optional boxes, including the `BitRateBox`, in the sample entry document the information for the output bitstream by the file parser reconstructed from this track and all the tracks this track depends on for decoding.

4.4 Video track structure

In the terminology of ISO/IEC 14496-12, 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`.

A video stream is represented by one or more video tracks in a file.

If there is more than one track representing scalable aspects of a single stream, then they form alternatives to each other, and the field '`alternate_group`' should be used, or the composition system used should select one of them, as appropriate. See ISO/IEC 14496-12:2015, 8.10.3 for informative labelling of why tracks are members of alternate groups.

4.5 Template fields used

The ISO base media file format defines a number of fields that have default values but that may be defined for use by specific sub-systems. Tracks containing video data may use the following template fields:

a)   `alternate_group` in the `TrackHeaderBox` (see subclause 5.4.8 on stream switching).

b)   template field '`depth`' in the VisualSampleEntry to document the presence of alpha.

   `depth` takes one of the following values:

   —   `0x18` – the video sequence is in colour with no alpha;

   —   `0x28` – the video sequence is in grayscale with no alpha;

   —   `0x20` – the video sequence has alpha (gray or colour).

4.6 Visual width and height

The width and height fields in a `VisualSampleEntry` shall correctly document the maximum cropped frame dimensions in any sample of the video 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.

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.

4.7 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, 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.

4.8 Sample groups on random access recovery points 'roll' and random access points 'rap '

This document disallows the presence of SampleToGroupBox with grouping_type equal to 'roll' or 'rap ' and with version greater than 0 in tracks with sample entries specified in this document other than Clause 7. Parsers according to this document shall ignore SampleToGroupBox with grouping_type equal to 'roll' or 'rap ' and with version greater than 0 in tracks with sample entries specified in this document other than Clause 7.

The video coding system can include the concept of a 'gradual decoding refresh' or random access recovery point. This may be signalled in the bitstream using a mechanism such as 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.

When all access units in output order starting from the access unit at the position of the SEI message can be successfully decoded after random access, i.e. when the recovery_frame_cnt syntax element of the recovery point SEI message is 0, the random access point ('rap ') sample grouping should be used.

This concept of gradual recovery is supported in the file format also by using RollRecoveryEntry groups. 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). In other words, RollRecoveryEntry groups can be used when the value of the recovery_frame_cnt syntax element of the recovery point SEI message is greater than 0.

NOTE   The roll-group counts samples in the file format; this does not necessarily 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 that can be decoded perfectly (either a sync sample, or the end of a pre-roll).

4.9 Hinting

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

Care should be taken when the structures in Annex A (aggregators or extractors) are in use and the track is hinted. These structures are defined only for use in the file format and should not be transmitted. In particular, a hint track that points at an extractor in a video track would cause the extractor itself to be transmitted (which is probably both incorrect and not the desired behaviour), not the data the extractor references. Hint tracks should normally directly reference NAL units specified in the applicable video coding standard.

4.10 On change of sample entry

In this clause, resetting decoding can be regarded as a similar process as starting the decoding from the beginning of a video stream.

A change in the sample entry is caused, for example, by a change of parameter set(s) reusing the same parameter set ID(s) in a `VisualSampleEntry`.

For the single-layer video file formats specified in subclause 4.11 and Clause 8, the video decoder for decoding of the bitstream output from the file parser is expected to be reset at the first sample at which the sample entry changes. When more than one sample entry is used in a track, random accessing is enabled if a video stream is encoded and encapsulated in the track as follows:

— The first sample that a sample entry is associated with (i.e., applies to) is a sync sample.

— Decoding of samples that a sample entry is associated with does not rely on data from any sample not associated with the sample entry or data from any other sample entry.

For the multi-layer video file formats specified in Clauses 6 and 7, the video decoder for decoding of the bitstream output from the file parser is expected to be reset at the first sample in the base track or scalable base track (as defined in Clauses 6 and 7) at which the sample entry changes. When more than one sample entry is used in the base track or scalable base track, random accessing is enabled if a video stream is encoded and encapsulated in the track as follows:

— The first sample that a sample entry in the base track or scalable base track is associated with is a sync sample.

— Decoding of the set of access units from the first access unit containing the first sample in the base track associated with a sample entry to the last access unit containing the last sample in the base track associated with the sample entry, inclusive, does not rely on data from any access unit earlier than the first access unit in decoding order or data from any sample entry associated with a sample contained in an access unit ealier than the first access unit in decoding order.

NOTE   When parameter sets are stored other than in the sample entry, i.e. either in a separate parameter set track, or in the samples in the video track, it is possible that there are two parameter sets with the same identifier, with the semantics (defined in the applicable video coding standard) that the second occurrence replaces the first from that time onwards in the stream. Converting such a stream to a form in which all parameter sets are in a single sample entry involves more than moving the applicable NAL units into the sample entry; one of the two occurrences will need a new identifier value, and the video NAL units that refer to it by identifier will also need updating. Alternatively, multiple sample entries can be used for the storage of parameter sets with same ID, in which case sample to sample entry association needs rewriting.

In the following, let the L-HEVC base track be the track referred to by a track reference of type `'oref'` (when present) or the present track (when it contains an `'oinf'` sample group). For the multi-layer video file formats specified in Clause 9, the video decoder for decoding of the bitstream output from the file parser is expected to be reset at the first sample in the L-HEVC base track at which the sample entry or the `'oinf'` sample group description entry changes. When more than one sample entry or `'oinf'` sample group description entry is used in the L-HEVC base track, random accessing is enabled if a video stream is encoded and encapsulated in the track as follows:

— The first sample that a sample entry or an `'oinf'` sample group description entry in the L-HEVC base track is associated with has an IRAP picture (as specified in ISO/IEC 23008-2) at the lowest layer carried in the track.

— Decoding of the set of access units from the first access unit containing the first sample in the L-HEVC base track associated with a sample entry or an `'oinf'` sample group description entry to the last access unit containing the last sample in the L-HEVC base track associated with the sample entry or `'oinf'` sample group description entry, inclusive, does not rely on data from any access unit earlier than the first access unit in decoding order or data from any sample entry or `'oinf'` sample group description entry associated with a sample contained in an access unit ealier than the first access unit in decoding order.

4.11 SEI information box

4.11.1 Definition

Box Type: 'seii'

Container: Scheme Information box ('schi') or VisualSampleEntry

Mandatory: Yes (in the SchemeInformationBox), No (in a VisualSampleEntry)

Quantity: One (in the SchemeInformationBox), Zero or one (in a VisualSampleEntry)

The SEI Information box documents the SEIs in a stream. When contained in a VisualSampleEntry numRequiredSEIs shall be 0. By inspecting the SEI Information box a player will know which SEI messages it can assume to be present, and which are deemed necessary by the file author for correct playback. There might be other SEIs present in the bitstream that are not documented by this box.

4.11.2 Syntax

```
aligned(8) class SeiInformationBox extends Box('seii') {

    unsigned int(16) numRequiredSEIs;

    for (i = 0; i < numRequiredSEIs; i++) {

    unsigned int(16) requiredSEI_ID;

    }

    unsigned int(16) numNotRequiredSEIs;

    for (i = 0; i < numNotRequiredSEIs; i++) {

    unsigned int(16) notrequiredSEI_ID;

    }

}
```

4.11.3 Semantics

requiredSEI_ID takes on the value "payloadType" of an SEI message present in the bitstream that is deemed necessary by the file author for correct playback.

notrequiredSEI_ID takes on the value "payloadType" of an SEI message present in the bitstream that is not deemed necessary by the file author for correct playback.

4.12 Post-decoder requirements scheme for signalling of SEI

4.12.1 General

In order to handle situations where the file author requires certain actions on the player or renderer, the ISO base media file format specifies the restricted-video mechanism where sample entries are hidden behind the generic sample entry 'resv'. The mechanism applies to all coding systems identified by chapters in this document. For the case of signalling of SEI, a file author can list occurring SEI message IDs (ISO/IEC 14496-10, ISO/IEC 23008-2) and classify them into two categories: those that are deemed required by the file author for correct playback, and others. The occurrence of either type of SEI messages can be signalled in the SEI Information box.

4.12.2 Definition

The scheme for signalling of SEI is defined here.

The SchemeType 'aSEI' is used.

The SEI information box is mandatory in the SchemeInformationBox under the 'aSEI' scheme. In this case, it contains information about the SEI messages present in the bitstream. Although the SEI messages are not required for decoding, the file author may require certain actions for rendering or other purposes. The box distinguishes between an SEI that is required to be understood for correct playback and an SEI that is not required for correct playback (but may enhance playback).

The SEI messages listed here should be stored either in the bitstream or in the Configuration Record. The SEI Information box does not contain the actual SEI messages, it only lists those that occur in the bitstream.

5 AVC elementary streams and sample definitions

5.1 Overview

ISO/IEC 14496-10 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, subclause 4.11 defines a storage format for video streams compressed using AVC.

This clause defines the storage for plain AVC streams, where 'plain AVC' refers to ISO/IEC 14496-10, excluding any multi-layer extension of ISO/IEC 14496-10 such as ISO/IEC 14496-10:2014, Annexes G, H, I and J.

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

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.

d) Switching pictures:

    to enable switching between different coded streams and substitution of pictures within the same stream.

e) 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).

f) 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.

5.2 Elementary stream structure

Two types of elementary streams are defined for storing AVC content (see also Figure 2):

— **A video elementary stream** contains 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 messages and access unit delimiter NAL units. Aggregators and Extractors, when present, shall not be directly output by file parsers.

— **Parameter set elementary streams** shall not contain video coding related NAL units (i.e. those NAL units containing video data or signalling video structure), and would normally contain only sequence parameter sets, picture parameter sets and sequence parameter set extension NAL units.

Using these stream types, AVC content shall be stored in one of these configurations:

— **Video elementary stream with no parameter sets**: In this case, sequence and picture parameter set NAL units shall be stored in the sample entries of this track. Sequence and picture parameter set NAL units shall not be part of AVC samples within the stream itself.

— **Video elementary stream possibly including parameter sets**: In this case, the sample entry indicates whether the stream may contain parameter sets of given types, in addition to other parameters provided in the sample entry. Sequence and picture parameter set NAL units may therefore 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 entries 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 2.

Table 2 — NAL unit types in elementary streams

Value of nal_unit_type	Description	Video elementary stream (sample entry 'avc1' or 'avc2')	Video elementary stream (sample entry 'avc3' or 'avc4')	Parameter set elementary stream
0	Unspecified	See Annex F	See Annex F	See Annex F
1	Coded slice of a non-IDR picture slice_layer_without_partitioning_rbsp( )	Yes	Yes	No
2	Coded slice data partition A slice_data_partition_a_layer_rbsp( )	Yes	Yes	No
3	Coded slice data partition B slice_data_partition_b_layer_rbsp( )	Yes	Yes	No
4	Coded slice data partition C slice_data_partition_c_layer_rbsp( )	Yes	Yes	No
5	Coded slice of an IDR picture slice_layer_without_partitioning_rbsp( )	Yes	Yes	No
6	Supplemental enhancement information(SEI) sei_rbsp( )	Yes Except for the sub-sequence, layering or filler SEI messages	Yes Except for the sub-sequence, or layering SEI messages	Only 'declarative' SEIs should be present
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 Parameter set elementary stream shall not be used	Yes

Table 2 *(continued)*

Value of `nal_unit_type`	Description	Video elementary stream (sample entry `'avc1'` or `'avc2'`)	Video elementary stream (sample entry `'avc3'` or `'avc4'`)	Parameter set elementary stream
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 Parameter set elementary stream shall not be used	Yes
9	Access unit delimiter (AU Delimiter) access_unit_delimiter_rbsp( )	Yes	Yes	No
10	End of sequence end_of_seq_rbsp()	Yes	Yes	No
11	End of stream end_of_stream_rbsp()	Yes	Yes	No
12	Filler data (FD) filler_data_rbsp( )	No	Yes	No
13	Sequence parameter set extension seq_parameter_set_extension_rbsp( )	No If parameter set elementary stream is not used, sequence parameter set extension shall be stored in the decoder specific information	Yes Parameter set elementary stream shall not be used	Yes
14…18	Reserved	Not specified by this document	Not specified by this document	Not specified by this document
19	Coded slice of an auxiliary coded picture without partitioning slice_layer_without_partitioning_rbsp( )	Yes	Yes	No
20…23	Reserved	Not specified by this document	Not specified by this document	Not specified by this document
24 – 31	Unspecified	See Annex F	See Annex F	See Annex F
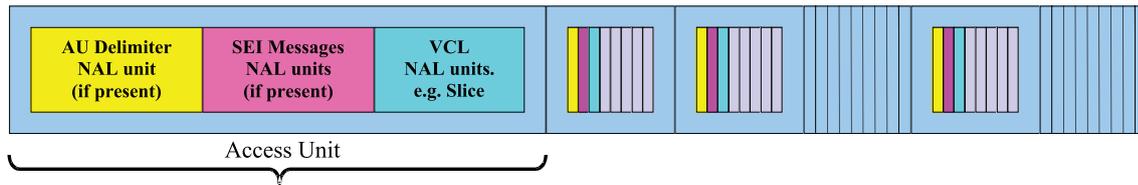
(a) Single video elementary stream containing NAL units
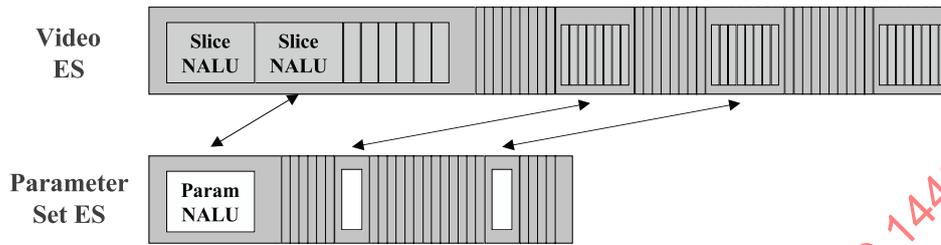
(b) Synchronized video and parameter sets with arrows denoting synchronization between streams

Figure 2 — AVC elementary stream structure

5.3 Sample and configuration definition

5.3.1 Overview

AVC sample: An AVC sample is an access unit as defined in ISO/IEC 14496-10:2014, 7.4.1.2.

AVC parameter set sample: An AVC parameter set sample is a sample in a parameter set stream that 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.

5.3.2 Canonical order and restrictions

The canonical stream format is an AVC elementary stream that satisfies the following conditions in addition to the general conditions in 4.3.2:

— **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 within the sample whose decoding time and composition time are those of the picture. Each sample shall contain at least one VCL NAL unit of the primary coded picture.

— **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 1   When the sample entry name is `'avc1'` or `'avc2'`, parameter sets are stored either in the sample entries 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 entries 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 can be inserted into the video elementary stream when the file is streamed over a transport that permits such parameter set updates. When the sample entry name is `'avc3'` or `'avc4'`, then parameter sets can be present both the sample entries and as part of the samples.

— **Parameter set track:** A sync sample in a parameter set track indicates that all parameter sets needed from that (decoding) 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 2   The use of a parameter set track in the file format does not require that a system delivering video content use a separate elementary stream for parameter sets. Instead, implementations can 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.

— **SEI message NAL units:** The order of SEI messages within a sample is as defined in ISO/IEC 14496-10:2014, 7.4.1.2.

— **Access unit delimiter NAL units:** The constraints obeyed by access unit delimiter NAL units are defined in ISO/IEC 14496-10:2014, 7.4.1.2.3.

— **Sub-sequence and layering SEI messages:** Sub-sequence or layering SEI messages should not occur in the AVC elementary stream. Specifically, the sub-sequence information, sub-sequence layer characteristics, and sub-sequence characteristics SEI messages should not occur in the stored AVC video elementary stream. Instead, all such information is stored as external metadata as described in subclause 5.4.7.

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

— **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 when the sample entry does not also permit parameter sets.

NOTE 4   The removal or addition of Filler Data NAL units, start codes, SEI messages or Filler Data SEI messages can change the bitstream characteristics with respect to conformance with the HRD when operating the HRD in CBR mode as specified in ISO/IEC 14496-10:2014, Annex C.

5.3.3   Decoder configuration information

5.3.3.1   AVC decoder configuration record

5.3.3.1.1   Definition

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 is supplied by the structure that contains it).

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

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

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).

When used to provide the configuration of a video elementary stream used without a parameter set elementary stream, the configuration record may or may not contain sequence or picture parameter sets (`numOfSequenceParameterSets` or `numOfPictureParameterSets` may or may not have the value 0).

The values for AVCProfileIndication, AVCLevelIndication, and the flags that indicate profile compatibility shall be valid for all parameter sets of the stream described by this record. The level indication shall 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 shall indicate a profile to which the entire stream associated with this configuration record 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 entire stream conforms to. If the entire stream is not examined, or the examination reveals that there is no profile to which the entire stream conforms, then the stream shall be split into two or more sub-streams with separate configuration records in which these rules are met.

Explicit indication can be provided in the AVC Decoder Configuration Record about the chroma format and bit depth used by the AVC video elementary stream. The parameter '`chroma_format_idc`' present in the sequence parameter set in AVC specifies the chroma sampling relative to the luma sampling. Similarly, the parameters '`bit_depth_luma_minus8`' and '`bit_depth_chroma_minus8`' in the sequence parameter set specify the bit depth of the samples of the luma and chroma arrays. The values of `chroma_format_idc`, `bit_depth_luma_minus8`' and '`bit_depth_chroma_minus8`' shall be identical in all sequence parameter sets in a single AVC configuration record. If two sequences differ in any of these values, two different AVC configuration records will be needed. If the two sequences differ in color space indications in their VUI information, then two different configuration records are also required.

The array of sequence parameter sets, and the array of picture parameter sets, may contain SEI messages of a 'declarative' nature, that is, those that provide information about the stream as a whole. An example of such an SEI is a user-data SEI. Such SEIs may also be placed in a parameter set elementary stream. NAL unit types that are reserved in ISO/IEC 14496-10 and in this document may acquire a definition in future, and readers should ignore NAL units with reserved values of NAL unit type when they are present in these arrays.

NOTE   This 'tolerant' behaviour is designed so that errors are not raised, allowing the possibility of backwards-compatible extensions to these arrays in future specifications.

When Sequence Parameter Set Extension NAL units occur in this record with AVCProfileIndication equal to 66, 77, or 88, they should be placed in the Sequence Parameter Set Array.

5.3.3.1.2 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;
   }
   if( AVCProfileIndication != 66 && AVCProfileIndication != 77 &&
      AVCProfileIndication != 88 )
   {
      bit(6) reserved = '111111'b;
      unsigned int(2) chroma_format;
      bit(5) reserved = '11111'b;
      unsigned int(3) bit_depth_luma_minus8;
      bit(5) reserved = '11111'b;
      unsigned int(3) bit_depth_chroma_minus8;
      unsigned int(8) numOfSequenceParameterSetExt;
      for (i=0; i< numOfSequenceParameterSetExt; i++) {
         unsigned int(16) sequenceParameterSetExtLength;
         bit(8*sequenceParameterSetExtLength) sequenceParameterSetExtNALUnit;
      }
   }
}
```

5.3.3.1.3 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 that 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.

`chroma_format` contains the chroma_format indicator as defined by the chroma_format_idc parameter in ISO/IEC 14496-10.

`bit_depth_luma_minus8` indicates the bit depth of the samples in the Luma arrays. For example, a bit depth of 8 is indicated with a value of zero (BitDepth = 8 + bit_depth_luma_minus8). The value of this field shall be in the range of 0 to 4, inclusive.

`bit_depth_chroma_minus8` indicates the bit depth of the samples in the Chroma arrays. For example, a bit depth of 8 is indicated with a value of zero (BitDepth = 8 + bit_depth_chroma_minus8). The value of this field shall be in the range of 0 to 4, inclusive.

`numOfSequenceParameterSetExt` indicates the number of Sequence Parameter Set Extensions that are used for decoding the AVC elementary stream.

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

`sequenceParameterSetExtNALUnit` contains a SPS Extension NAL unit, as specified in ISO/IEC 14496-10.

5.4 Derivation from ISO base media file format

5.4.1 AVC file type and identification

Conformance with this document 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; in a file with extension ".mp4", the major brand may be `'avc1'`.

Readers conformant to this document 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.

5.4.2   AVC video stream definition

5.4.2.1   Sample entry name and format

5.4.2.1.1   Definition

Sample Entry and Box Types: 'avc1', 'avc2', 'avc3', 'avc4', 'avcC', 'm4ds', 'btrt'

Container: Sample Description Box ('stsd')

Mandatory: An 'avc1', 'avc2', 'avc3' or 'avc4' sample entry 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 5.3.3.1.

An optional BitRateBox 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 entries may be used, as permitted by ISO/IEC 14496-12, to indicate sections of video that use different configurations or parameter sets.

The sample entry name 'avc1' or 'avc3' may only be used when the stream to which this sample entry applies is a compliant and AVC stream as viewed by an AVC decoder operating under the configuration (including profile and level) given in the AVCConfigurationBox. The file format specific structures that resemble NAL units (defined in Annex A) may be present but shall not be used to access the AVC base data; that is, the AVC data shall not be contained in Aggregators (though they may be included within the bytes referenced by the additional_bytes field) nor referenced by Extractors.

The sample entry name 'avc2' or 'avc4' may only be used when Extractors or Aggregators (Annex A) are required to be supported, and an appropriate Toolset is required (for example, as indicated by the file-type brands). This sample entry type indicates that, in order to form the intended AVC stream, Extractors shall be replaced with the data they are referencing, and Aggregators shall be examined for contained NAL Units. Tier grouping may be present.

5.4.2.1.2   Syntax

```
   // Visual Sequences
class AVCConfigurationBox extends Box('avcC') {
   AVCDecoderConfigurationRecord() AVCConfig;
}
class MPEG4ExtensionDescriptorsBox extends Box('m4ds') {
   Descriptor Descr[0 .. 255];
}
class AVCSampleEntry() extends VisualSampleEntry (type) {
                            // type is 'avc1' or 'avc3'
   AVCConfigurationBox  config;
   MPEG4ExtensionDescriptorsBox (); // optional
}
class AVC2SampleEntry() extends VisualSampleEntry (type) {
                            // type is 'avc2' or 'avc4'
   AVCConfigurationBox  avcconfig;
   MPEG4ExtensionDescriptorsBox descr;  // optional
}
```

5.4.2.1.3   Semantics

Compressorname in the base class VisualSampleEntry indicates the name of the compressor used with the value "\012AVC Coding" being recommended; the first byte is a count of the remaining bytes, here represented by \012, which (being octal 12) is 10 (decimal), the number of bytes in the rest of the string.

config is defined in subclause 5.3.3. If a separate parameter set stream is used, numOfSequenceParameterSets and numOfPictureParameterSets shall both be zero.

Descr is a descriptor that 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.

5.4.3   AVC parameter set stream definition

5.4.3.1   Sample entry name and format

5.4.3.1.1   Definition

Sample Entry Type: 'avcp'

Container: Sample Description Box ('stsd')

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.

5.4.3.1.2   Syntax

```
class AVCParameterSampleEntry() extends VisualSampleEntry ('avcp'){
```

```
   AVCConfigurationBox  config;

}
```

5.4.3.1.3   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 subclause 5.3.3. `numOfSequenceParameterSets` and `numOfPictureParameterSets` shall both be zero.

5.4.3.2   Sample format

5.4.3.2.1   Definition

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

5.4.3.2.2   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;
   }
}
```

5.4.3.2.3   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.

5.4.3.3   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.

5.4.4   Parameter sets

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

Each AVC sample entry, 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.

When the sample entry name is `'avc1'` or `'avc2'`, the following applies:

— In the file format each configuration of parameter sets is represented separately. A parameter set cannot be updated without causing a different sample entry 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 entry 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 entry must change to a second sample entry, 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 entry 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   When several parameter sets are used and parameter set updating is desired, a separate parameter set elementary stream would be used.

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

When the sample entry name is `'avc3'` or `'avc4'`, parameter sets may be present in both sample entry and as part of samples, and an update of a parameter set by a parameter set of the same type that is stored as part of a sample is possible.

5.4.5   Sync sample

A sample is considered as a sync sample if the video data NAL units in the sample indicate that the primary coded picture contained in the sample is an instantaneous decoding refresh (IDR) picture.

When the sample entry name is `'avc1'` or `'avc2'`, all SPSs and PPSs needed to decode the video data NAL units in the sample of the IDR picture and the following samples in decode order are contained in the decoder configuration of the video elementary stream or in a separate parameter set elementary stream sample.

When the sample entry name is `'avc3'` or `'avc4'`, the following applies:

1.   If the sample is a sync sample, all parameter sets needed for decoding the sample shall be included either in the sample entry or in the sample itself.

2.   Otherwise (the sample is not a sync sample), all parameter sets needed for decoding the same shall be included either in the sample entry or in any of the samples since the previous sync sample to the sample itself, inclusive.

A parameter set elementary stream sample is a sync sample 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.

5.4.6   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 the general requirements and for the specific coding format in the track.

While the use of shadow sync is supported for backward compatibility reasons, this use is deprecated and use of the mechanisms defined in subclause 5.4.8 is recommended.

5.4.7 Layering and sub-sequences

5.4.7.1 Overview

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 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 ISO/IEC 14496-10:2014, Annex D. 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 SampleToGroupBox (per track) with a grouping_type equal to 'avll'. This SampleToGroupBox 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 SampleToGroupBox (per track) with a grouping_type equal to 'avss'. This SampleToGroupBox 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.

5.4.7.2 Sub-sequence description entry

5.4.7.2.1 Definition

Group Type: '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 depends 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 (SampleToGroupBox, SampleGroupDescriptionBox) per track concerning the grouping of samples into layers and sub-sequences.

The grouping_type_parameter is not defined for the SampleToGroupBox with grouping type 'avss'.

5.4.7.2.2 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(7) reserved = 0;
         unsigned int(1) accurateStatisticsFlag;
         unsigned int(16) avgBitRate;
         unsigned int(16) avgFrameRate;
      }
      unsigned int(8) numReferences;
      DependencyInfo dependency[numReferences];
   }
}
```

5.4.7.2.3 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 that 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 that 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 ÷ (($t_2 - t_1$) * 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.

5.4.7.3   Layer description entry

5.4.7.3.1   Definition

Group Type: `'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.

The `grouping_type_parameter` is not defined for the `SampleToGroupBox` with grouping type `'avll'`.

5.4.7.3.2   Syntax

```
class AVCLayerEntry() extends VisualSampleGroupEntry ('avll')

{

    unsigned int(8)  layerNumber;

    unsigned int(7)  reserved = 0;

    unsigned int(1)  accurateStatisticsFlag;

    unsigned int(16) avgBitRate;

    unsigned int(16) avgFrameRate;

}
```

5.4.7.3.3   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.

5.4.8   Alternate streams and switching pictures

5.4.8.1   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 may 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. The following figure shows how SP pictures are used to switch between two different bit streams.
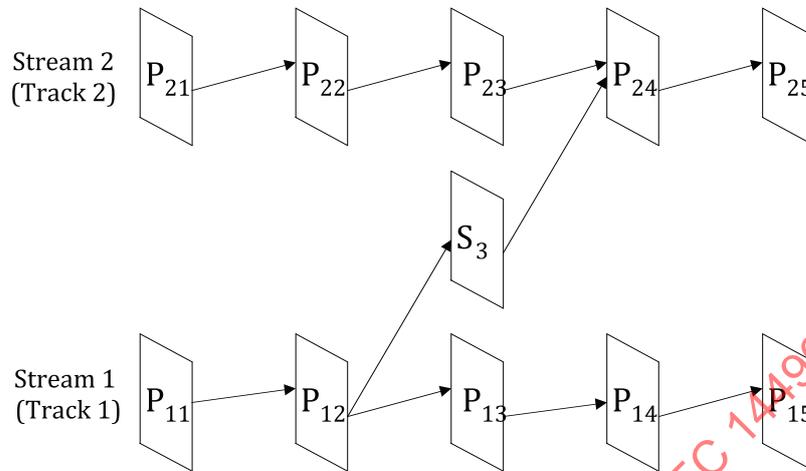
Figure 3 — Stream switching

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 shall 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.

5.4.8.2   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 bitstream can be stored as a separate track and related together as alternate tracks. All the tracks that form a group that may be switched between, but not the tracks containing the switching pictures, shall 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.

5.4.8.3   Track references

The switching track shall 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 shall be linked to the track from which it switches (the source track) by a track reference of type 'swfr' in the switching picture track.

5.4.8.4   Sample dependency

5.4.8.4.1   Definition

Box Type: 'sdep'

Container: Sample Table 'stbl'

Mandatory: No

Quantity: Zero or exactly one.

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.

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').

5.4.8.4.2   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;

        }

    }

}
```

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

5.4.9   Definition of a sub-sample for AVC

For the use of the sub-sample information box (ISO/IEC 14496-12:2015, 8.7.7) in an AVC stream, a sub-sample is defined as one or more contiguous NAL units within a sample and having the same value of the following fields; `RefPicFlag`, `RedPicFlag` and `VclNalUnitFlag`. Each sub-sample includes both NAL unit(s) and their preceding NAL unit length field(s). The presence of this box is optional; however, if present in a track containing AVC data, it shall have the semantics defined here.

The subsample_priority field shall be set to a value in accordance with the specification of this field in ISO/IEC 14496-12.

The discardable field shall be set to 1 only if this sample is still decodable if this sub-sample is discarded (e.g. the sub-sample consists of an SEI NAL unit, or a redundant coded picture).

The `codec_specific_parameters` field of the Subsample Information box is defined for AVC as follows:

```
unsigned int(1) RefPicFlag;

unsigned int(1) RedPicFlag;

unsigned int(1) VclNalUnitFlag;

bit(29) reserved = 0;
```

`RefPicFlag` equal to 0 indicates that all the NAL units in the sub-sample have `nal_ref_idc` equal to 0. `RefPicFlag` equal to 1 indicates that all the NAL units in the sub-sample have `nal_ref_idc` greater than 0.

`RedPicFlag` equal to 0 indicates that all the NAL units in the sub-sample have `redundant_pic_cnt` equal to 0. `RedPicFlag` equal to 1 indicates that all the NAL units in the sub-sample have `redundant_pic_cnt` greater than 0.

`VclNalUnitFlag` equal to 0 indicates that all NAL units in the sub-sample are non-VCL NAL units. Value 1 indicates that all NAL units in the sub-sample are VCL NAL units.

6   SVC elementary stream and sample definitions

6.1   Overview

This clause specifies the storage format of SVC data. It extends the definitions of the storage format of AVC in subclause 4.11.

The file format for storage of SVC content, as defined in this clause and Annexes A to D, uses the existing capabilities of the ISO base media file format and the plain AVC file format (i.e. the file format specified in subclause 4.11). In addition, the following new extensions, among others, to support SVC-specific features are specified.

a)   Scalable or multiview grouping:

a structuring and grouping mechanism to indicate the association of NAL units with different types and hierarchy levels of scalability.

b)   Aggregator:

a structure to enable efficient scalable grouping of NAL units by changing irregular patterns of NAL units into regular patterns of aggregated data units.

c)   Extractor:

a structure to enable efficient extraction of NAL units from other tracks than the one containing the media data.

d) Temporal metadata statements:

structures for storing time-aligned information of media samples.

e) AVC compatibility:

a provision for storing an SVC bitstream in an AVC compatible manner, such that the AVC compatible base layer can be used by any plain AVC file format compliant reader.

6.2 Elementary stream structure

SVC streams are stored in accordance with 5.2, with the following definition of an SVC video elementary stream:

— **An SVC Video Elementary Streams** contains all video coding related NAL units (i.e. those NAL units containing video data or signalling video structure, possibly after resolution of extractors and aggregators) and may contain non-video coding related NAL units such as SEI messages and access unit delimiter NAL units. Aggregators and Extractors, when present, shall not be directly output by file parsers.

SVC streams may also be stored using associated parameter set streams, if needed.

For SVC streams, Table 2 is updated as follows in Table 3; only entries where the definition for SVC differs from AVC, are shown.

Table 3 — NAL unit types in SVC and AVC streams

Value of nal_unit_type	Description	AVC video elementary stream	SVC video elementary stream (sample entry 'avc1', 'avc2', or 'svc1')	SVC video elementary stream (sample entry 'avc3', 'avc4', or 'svc2')	Parameter set elementary stream
14	Prefix NAL unit in scalable extension prefix_nal_unit_rbsp( )	Not specified	Yes	Yes	No
15	Subset sequence parameter set subset_seq_parameter_set_rbsp( )	Not specified	No. If parameter set elementary stream is not used, Subset SPS shall be stored in the Decoder Specific Information.	Yes. Parameter set elementary stream shall not be used.	Yes
20	Coded slice in scalable extension slice_layer_ extension_rbsp( )	Not specified	Yes	Yes	No
24 – 31	Not specified	See Annex F	See Annex F	See Annex F	See Annex F

NOTE   slice_layer_extension_rbsp was previously called slice_layer_in_scalable_extension_rbsp.

There may be AVC VCL NAL units, SVC VCL NAL units and other NAL units, i.e. non-VCL NAL units, present in an SVC video elementary stream. Additionally, there may be Aggregators and Extractors present in an SVC video elementary stream.

An AVC VCL NAL unit in an SVC video elementary stream conforming to one or more profiles specified in ISO/IEC 14496-10:2014, Annex G shall be immediately preceded by a prefix NAL unit containing the scalability information for the AVC VCL NAL unit. In this file format an AVC VCL NAL unit and the

immediately preceding prefix NAL unit are logically seen as one NAL unit: the prefix NAL unit provides the scalability information and the AVC VCL NAL unit provides the NAL unit type and payload.

6.3   Use of the plain AVC file format

The SVC file format is an extension of the plain AVC file format defined in subclause 4.11.

Subclause 5.4.7 is defined for use with plain AVC streams. Its use with SVC streams is deprecated.

6.4   Sample and configuration definition

6.4.1   Overview

SVC Sample: An SVC sample consists of the NAL units that belong to an access unit as defined in ISO/IEC 14496-10:2014, 7.4.1.2 and that are represented by the track.

6.4.2   Canonical order and restrictions

6.4.2.1   Restrictions

The following restrictions apply to SVC data in addition to the requirements in subclause 5.3.2.

— **SVC coded slice NAL units** (Coded slices in scalable extension): All SVC coded slice NAL units for a single instant in time shall be contained in the sample whose composition time is that of the picture represented by the access unit. An SVC sample shall contain at least one AVC or SVC VCL NAL unit.

— **Prefix NAL units** (Prefix NAL unit in scalable extension): Each prefix NAL unit is placed immediately before the corresponding AVC VCL NAL unit, providing scalability information about the AVC VCL NAL unit.

  NOTE   Prefix NAL units can also be associated with filler data NAL units.

— **Aggregators/Extractors**: The order of all NAL units included in an Aggregator or referenced by an Extractor is exactly the decoding order as if these NAL units were present in a sample not containing aggregators or extractors. After processing the Aggregator or the Extractor, all NAL units must be in valid decoding order as specified in ISO/IEC 14496-10.

6.4.2.2   Decoder configuration record

When the decoder configuration record defined in subclause 5.3.3.1 is used for a stream that can be interpreted as either an SVC or AVC stream, the AVC decoder configuration record shall reflect the properties of the AVC compatible base layer, e.g. it shall contain only parameter sets needed for decoding the AVC base layer.

If the sample entry name is 'svc1', a parameter set stream may be used with SVC streams, as with AVC streams, in which case, parameter sets shall not be included in the decoder configuration record. Otherwise (the sample entry name is 'svc2'), parameter sets may be stored in both the decoder configuration record or as part of samples while a parameter set elementary stream shall not be used.

Sequence or picture parameter sets are numbered in order of storage from 1 to numOfSequenceParameterSets or numOfPictureParameterSets respectively. Sequence and Picture parameter sets stored in this record in a file may be referenced using this 1-based index by the InitialParameterSetBox.

The SVCDecoderConfigurationRecord is structurally identical to an AVCDecoderConfigurationRecord. However, the reserved bits preceding and succeeding the lengthSizeMinusOne field are re-defined. The syntax is as follows:

```
aligned(8) class SVCDecoderConfigurationRecord {

   unsigned int(8) configurationVersion = 1;

   unsigned int(8) AVCProfileIndication;

   unsigned int(8) profile_compatibility;

   unsigned int(8) AVCLevelIndication;

   unsigned int(1) complete_representation;

   bit(5) reserved = '11111'b;

   unsigned int(2) lengthSizeMinusOne;

   bit(1) reserved = '0'b;

   unsigned int(7) 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;

   }

}
```

The semantics of the fields `AVCProfileIndication`, `profile_compatibility`, and `AVCLevelIndication` differ from the `AVCDecoderConfigurationRecord` as follows:

The fields `AVCProfileIndication`, `AVCLevelIndication` carry the profile and level indications, respectively, indicating the profile and level of the entire scalable stream in this track. They, and the `profile_compatibility` field, shall have values such that an SVC decoder supporting these values is able to decode bitstreams conforming to the profile, level and profile compatibility flags indicated in any of the sequence parameter sets or subset sequence parameter sets contained in this record.

The semantics of other fields are as follows, or are as defined for an `AVCDecoderConfigurationRecord`:

—   `complete_representation` is set on a minimal set of tracks that contain a portion of the original encoded scalable stream, as defined in subclause 6.5.1. Other tracks may be removed from the file without loss of any portion of the original encoded bitstream, and, once the set of tracks has been reduced to only those in the complete subset, any further removal of a track removes a portion of the encoded information.

—   `numOfSequenceParameterSets` indicates the number of SPSs and subset SPSs that are used for decoding the SVC elementary stream. The value of `numOfSequenceParameterSets` shall be in the range of 0 to 64, inclusive.

—   `SequenceParameterSetLength` indicates the length in bytes of the SPS or subset SPS NAL unit.

— `SequenceParameterSetNALUnit` contains a SPS or subset SPS NAL unit. SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Subset SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Any SPS shall occur before all the subset SPSs, if any.

6.5 Derivation from the ISO base media file format

6.5.1 SVC track structure

A scalable video stream is represented by one or more video tracks in a file. Each track represents one or more operating points of the scalable stream. A scalable stream may, of course, be further thinned, if desired.

There is a minimal set of one or more tracks that, when taken together, contain the complete set of encoded information. All these tracks shall have the flag "`complete_representation`" set in all their sample entries. This group of tracks that form the complete encoded information are called the "complete subset".

Let the lowest operating point be the one of all the operating points represented by DTQ (dependency_id, temporal_id and quality_id) combinations that has the least values of dependency_id, temporal_id and quality_id, respectively. The track that has the flag "`complete_representation`" set and contains the lowest operating point shall be nominated as the 'scalable base track'. All the other tracks that are part of the same scalable encoded information shall be linked to this base track by means of a track reference of type `'sbas'` (scalable base). The complete encoded information can be retained when the tracks included in the "complete subset" are retained; all other tracks shall represent subsets, copies (i.e. the same media data) or re-orderings of the complete subset.

NOTE 1   An alternate group can also include completely independent bitstreams, as well as alternative operating points of the same bitstream. The SVC tracks in the alternate group shall be examined to see how many scalable base tracks are identified.

NOTE 2   "A scalable bitstream" can require more than one track to represent it (consider a stream with a low-resolution, low-frame-rate base layer, and a high resolution enhancement layer, and a high frame-rate enhancement layer, but missing the data for high resolution high frame-rate). However, such a scalable bitstream is typically a non-conforming bitstream.

All the tracks sharing the same scalable base track shall share the same timescale as the scalable base track.

6.5.2 Data sharing and extraction

Different tracks may logically share data. This sharing can take one of the following two forms:

a)   The sample data is copied (i.e., the same media data is present) in different tracks.

b)   There may be instructions on how to perform this copy at the time that the file is read.

For the second case, Extractors (defined in A.3) are used.

6.5.3 SVC video stream definition

6.5.3.1 Sample entry name and format

6.5.3.1.1 Definition

Sample Entry and Box Types: 'svc1', 'svc2', 'svcC', 'seib'

Container: Sample Description Box ('stsd')

Mandatory: One of the 'avc1', 'avc2', 'avc3', 'avc4',

'svc1', and 'svc2' sample entries is mandatory.

Quantity: One or more sample entries may be present

If an SVC elementary stream contains an AVC compatible base layer, then an AVC visual sample entry ('avc1', 'avc2', 'avc3', or 'avc4') shall be used. Here, the entry shall contain initially an AVC Configuration Box, possibly followed by an SVC Configuration Box as defined below. The AVC Configuration Box documents the Profile, Level and Parameter Set information pertaining to the AVC compatible base layer as defined by the AVCDecoderConfigurationRecord. The SVC Configuration Box documents the Profile, Level, and possibly also parameter sets pertaining to the entire stream containing the SVC compatible enhancement layers as defined by the SVCDecoderConfigurationRecord, stored in the SVCConfigurationBox.

If the SVC elementary stream does not contain an AVC base layer, then an SVC visual sample entry ('svc1' or 'svc2') shall be used. The SVC visual sample entry shall contain an SVC Configuration Box, as defined below. This includes an SVCDecoderConfigurationRecord, as defined in this document.

The lengthSizeMinusOne field in the SVC and AVC configurations in any given sample entry shall have the same value.

A priority assignment URI provides the name (in the URI space) of a method used to assign priority_id values. When it occurs in an AVC or SVC sample entry, exactly one URI shall be present, that documents the priority_id assignments in the stream. The URI is treated here as a name only; it should be de-referenceable, though this is not required. File readers may be able to recognize some methods and thereby know what stream extraction operations based on priority_id would do.

Extractors or aggregators may be used for SVC VCL NAL units in 'avc1', 'avc2', 'avc3', 'avc4', 'svc1' or 'svc2' tracks.

NOTE   When AVC compatibility is indicated, it can be necessary to indicate an unrealistic level for the AVC base layer, to accommodate the bit rate of the entire stream, because all the NAL units are considered as included in the AVC base layer and hence can be fed to the decoder, which is expected to discard those NAL unit it does not recognize. This case happens when the 'avc1' or 'avc3' sample entry is used and both AVC and SVC configurations are present.

Either or both of a ScalabilityInformationSEIBox or SVCConfigurationBox may be present in an 'avc1' or 'avc3' sample entry. In this case the AVCSVCSampleEntry definition below applies.

The parameter sets required to decode a NAL unit that is present in the sample data of a video stream, either directly or by reference from an Extractor, shall be present in the decoder configuration of that video stream or in the associated parameter set stream (if used).

Table 4 shows for a video track all the possible uses of sample entries, configurations and the SVC tools (excluding timed metadata, which is always used in another track):

Table 4 — Use of sample entries for AVC and SVC tracks

sample entry name	with configuration records	meaning
`'avc1'` or `'avc3'`	AVC Configuration Only	A plain AVC track with AVC NAL units only; Extractors, aggregators, and tier grouping shall not be present.
`'avc1'` or `'avc3'`	AVC and SVC Configurations	An SVC track with both AVC and SVC NAL units; Extractors and aggregators shall not be present.
`'avc2'` or `'avc4'`	AVC Configuration Only	A plain AVC track with AVC NAL units only; Extractors may be present and used to reference AVC NAL units; Aggregators may be present to contain and reference AVC NAL units; Tier grouping may be present.
`'avc2'` or `'avc4'`	AVC and SVC Configurations	An SVC track with both AVC and SVC NAL units; Extractors and aggregators may be present; Extractors may reference both AVC and SVC NAL units; Aggregators shall not contain but may reference AVC NAL units, and may both contain and reference SVC NAL units; Tier grouping may be present.
`'svc1'` or `'svc2'`	SVC Configuration	An SVC track without AVC NAL units; Extractors shall be present to reference AVC NAL units (from a different track) and may also be used to reference SVC NAL units; Aggregators may be present to contain and reference SVC NAL units; Tier grouping may be present.

6.5.3.1.2　Syntax

```
class SVCConfigurationBox extends Box('svcC') {

    SVCDecoderConfigurationRecord() SVCConfig;

}

class ScalabilityInformationSEIBox extends Box('seib', size)

{

    unsigned int(8*size-64) scalinfosei;

}

class SVCPriorityAssignmentBox extends Box('svcP')

{

    unsigned int(8)   method_count;

    string PriorityAssignmentURI[method_count];

}

class AVCSVCSampleEntry() extends AVCSampleEntry ('avc1' or 'avc3') {

    SVCConfigurationBox   svcconfig;       // optional

    ScalabilityInformationSEIBox   scalability;   // optional

    SVCPriorityAssignmentBox   method;           // optional

}

class AVC2SVCSampleEntry() extends AVC2SampleEntry('avc2' or 'avc4') {
```

```
    SVCConfigurationBox  svcconfig;      // optional

    ScalabilityInformationSEIBox  scalability;  // optional

    SVCPriorityAssignmentBox  method;        // optional

}

// Use this if the track is NOT AVC compatible

class SVCSampleEntry() extends VisualSampleEntry ('svc1' or 'svc2') {

    SVCConfigurationBox     svcconfig;

    MPEG4ExtensionDescriptorsBox descr;  // optional

    ScalabilityInformationSEIBox  scalability;  // optional

    SVCPriorityAssignmentBox  method;        // optional

}
```

6.5.3.1.3   Semantics

When the sample entry is 'svc1' or 'svc2', Compressorname in the base class VisualSampleEntry indicates the name of the compressor used, with the value "\012SVC Coding" being recommended (\012 is 10, the length of the string "SVC coding" in bytes).

scalinfosei contains an SEI NAL unit containing only a scalability information SEI message as specified in ISO/IEC 14496-10:2014, Annex G. The 'size' field of the container box ScalabilityInformationSEIBox shall not be equal to 0 or 1.

method_count provides a count of the number of following URIs. This field shall take the value 1 in an 'avc1','avc2','avc3','avc4','svc1' or 'svc2' sample entry.

PriorityAssignmentURI provides a unique name of the method used to assign priority_id values. In the case of absence of this box, the priority assignment method is unknown.

6.5.4   SVC visual width and height

The visual width and height documented in a VisualSampleEntry of a stream containing SVC VCL NAL unit are the cropped frame dimensions of the AVC base layer, if the stream is described by a sample entry of type 'avc1', 'avc2', 'avc3' or 'avc4'; otherwise they are the cropped frame dimensions of decoded pictures by decoding the entire stream.

6.5.5   Sync sample

For video data described by a sample entry of type 'avc1', 'avc2', 'avc3', or 'avc4', the sync sample table identifies IDR access units for both an AVC decoder, and an SVC decoder (if any) operating on the entire bitstream.

For video data described by a sample entry of type 'svc1', the sync sample table identifies IDR access units in the entire SVC bitstream.

For video data described by a sample entry of type 'svc2', the sync sample table identifies IDR access units in the entire SVC bitstream, and additionally the following applies:

1. If the sample is a sync sample, all parameter sets needed for decoding that sample shall be included either in the sample entry or in the sample itself.

2. Otherwise (the sample is not a sync sample), all parameter sets needed for decoding the sample shall be included either in the sample entry or in any of the samples since the previous sync sample to the sample itself, inclusive.

NOTE   The sync sample table, if present, documents only access units that are IDR access units for both the AVC compatible base layer and the layer corresponding to decoding the entire bitstream contained in the track. In case documenting of layer-specific IDR access units is desired, the stream should be stored in separate tracks, e.g. two tracks, one containing the AVC base layer with a sample entry of type 'avc1' or 'avc3', and the other containing the SVC enhancement layers with a sample entry of type 'svc1' or 'svc2'. However, extractors shall then be used for tracks that are not the scalable base track.

6.5.6   Shadow sync

A shadow sync box shall not be used for video data described by an 'svc1' or 'svc2' sample entry. Its use for SVC is deprecated.

6.5.7   Independent and disposable samples box

If the SampleDependencyTypeBox is used in a track that is both AVC and SVC compatible, then care should be taken that the information provided by this box is true no matter what valid subset of the SVC data (possibly only the AVC data) is used. The 'unknown' values (value 0 of the fields sample_depends_on, sample_is_depended_on, and sample_has_redundancy) may be needed if the information varies.

6.5.8   Sample groups on random access recovery points 'roll' and random access points 'rap '

For video data described by a sample entry of type 'avc1', 'avc2', 'avc3' or 'avc4', the random access recovery sample group and the random access point sample group identify random access recovery points and random access points, respectively, for both an AVC decoder, and an SVC decoder (if any) operating on the entire bitstream.

NOTE   If the random access recovery points or the random access points for the AVC decoder and the SVC decoder operating on the entire bitstream are not all aligned, the random access recovery points table or the random access point table, respectively, will not document all of them. In this case, the stream can be stored in multiple tracks, e.g. two tracks, one containing the AVC base layer with a sample entry of type 'avc1' or 'avc3', and the other containing the SVC enhancement layers with a sample entry of type 'svc1' or 'svc2'.

For video data described by a sample entry of type 'svc1' or 'svc2', the information provided by the random access recovery sample group and the random access point sample group is true for any valid subset of the entire SVC bitstream.

6.5.9   Definition of a sub-sample for SVC

This subclause extends the definition of a sub-sample for AVC in 5.4.9.

For the use of the sub-sample information box (ISO/IEC 14496-12:2015, 8.7.7) in an SVC stream, a sub-sample is defined as one or more contiguous whole NAL units having the same values of the following fields: RefPicFlag, RedPicFlag, VclNalUnitFlag, IdrFlag, PriorityId, DependencyId, QualityId, TemporalId, UseRefBasePicFlag, DiscardableFlag and StoreBaseRepFlag, specified subsequently. Each sub-sample includes both NAL unit(s) and their preceding NAL unit length field(s). The presence of this box is optional; however, if present in a track containing SVC data, it shall have the semantics defined here.

As required in 5.4.9, the subsample_priority field shall be set to a value in accordance with the specification of this field in ISO/IEC 14496-12.

The codec_specific_parameters field of the Subsample Information box is defined for SVC as follows:

```
unsigned int(1) RefPicFlag;

unsigned int(1) RedPicFlag;

unsigned int(1) VclNalUnitFlag;
```

```
bit(6) reserved = 0;

unsigned int(1) IdrFlag;

unsigned int(6) PriorityId;

bit(1) reserved = 0; // corresponding to no_inter_layer_pred_flag

unsigned int(3) DependencyId;

unsigned int(4) QualityId;

unsigned int(3) TemporalId;

unsigned int(1) UseRefBasePicFlag;

unsigned int(1) DiscardableFlag;

bit(1) reserved = 0; // corresponding to output_flag

unsigned int(1) StoreBaseRepFlag;

bit(1) reserved = 0;
```

For an AVC VCL NAL unit in an SVC context, the prefix NAL unit shall be grouped with the AVC VCL NAL unit in the same sub-sample, and its fields values apply to the AVC VCL NAL unit.

RefPicFlag equal to 0 indicates that all the NAL units in the sub-sample have nal_ref_idc equal to 0. RefPicFlag equal to 1 indicates that all the NAL units in the sub-sample have nal_ref_idc greater than 0.

RedPicFlag equal to 0 indicates that all the NAL units in the sub-sample have redundant_pic_cnt equal to 0. RedPicFlag equal to 1 indicates that all the NAL units in the sub-sample have redundant_pic_cnt greater than 0.

VclNalUnitFlag equal to 0 indicates that all NAL units in the sub-sample are non-VCL NAL units. Value 1 indicates that all NAL units in the sub-sample are VCL NAL units.

IdrFlag indicates the idr_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of idr_flag.

PriorityId indicates the priority_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of priority_id.

NoInterLayerPredFlag indicates the no_inter_layer_pred_flag of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of no_inter_layer_pred_flag.

DependencyId indicates the dependency_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same dependency_id value.

QualityId indicates the quality_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same quality_id value.

TemporalId indicates the temporal_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same temporal_id value.

UseRefBasePicFlag indicates the use_ref_base_pic_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of use_ref_base_pic_flag.

DiscardableFlag indicates the discardable_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same discardable_flag value.

NOTE   This is not the same definition as the discardable field in the sub-sample information box.

`StoreBaseRepFlag` indicates the store_base_rep_flag value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same value of store_base_rep_flag.

7   MVC and MVD elementary stream and sample definitions

7.1   Overview

This clause specifies the storage format of MVC data. It extends the definitions of the storage format of AVC in subclause 4.11.

The file format for storage of MVC and MVD content, as defined in this clause and Annexes A to D uses the existing capabilities of the ISO base media file format and the plain AVC file format (i.e. the file format specified in subclause 4.11). In addition, the following new extensions, among others, to support MVC- and MVD-specific features are specified.

a)   Multiview grouping:

   a structuring and grouping mechanism to indicate the association of NAL units with different types and hierarchy levels of scalability.

b)   Aggregator:

   a structure to enable efficient scalable grouping of NAL units by changing irregular patterns of NAL units into regular patterns of aggregated data units.

c)   Extractor:

   a structure to enable efficient extraction of NAL units from other tracks than the one containing the media data.

d)   Temporal metadata statements:

   structures for storing time-aligned information of media samples.

e)   AVC compatibility:

   a provision for storing an MVC or MVD bitstream in an AVC compatible manner, such that the AVC compatible base layer can be used by any plain AVC file format compliant reader.

The support for MVC or MVD includes a number of tools, and there are various 'models' of how they might be used. In particular, an MVC or MVD stream can be placed in tracks in a number of ways, among which are the following:

1.   all the views in one track, labelled with sample groups;

2.   each view, including both texture views and depth views when both are present, in its own track, labelled in the sample entries;

3.   a hybrid, one track containing all views, and one or more single-view tracks each containing a view that can be independently coded;

4.   the expected operating points each in a track (e.g. the AVC base, a stereo pair, a multiview scene, or an MVD scene).

5.   (for MVD only) each texture or depth view in its own track, labelled in the sample entries.

The MVC and MVD file format allows storage of one or more views into a track, similarly to the support for SVC in Clause 6. Storage of multiple views per track can be used, e.g., when a content provider wants to provide a multiview bitstream that is not intended for subsetting or when the bitstream has been created for a few pre-defined sets of output views (such as 1, 2, 5, or 9 views) where tracks can be created accordingly. If more than one view is stored in a track and there are several tracks (more than one)

representing the MVC or MVD bitstream, the use of the sample grouping mechanism is recommended. The sample grouping mechanism is used to define tiers identifying the views present in the track and to extract required NAL units for certain operating points conveniently. The sample grouping mechanism is usually used with aggregators to form regular NAL unit patterns within samples. Thus, SVC-like sample grouping, aggregators, and view definitions for sample groups are specified for MVC or MVD.

The Multiview Information box ('mvci') is specified to indicate information that applies to more than one view, such as the target output views in one or more Multiview Group boxes. Characteristics (such as camera parameters) of the respective bitstream subset can also be indicated within the Multiview Group box using the Multiview Relation Attributes box ('mvra'), which is similar to the Track Selection box.

A player should have means to determine which views are preferred for displaying, and select one or more tracks that provide the data for the desired operating point, preferring a track that is specific to that operating point over tracks that also contain other data. The display characteristics of players may differ; for example, the number of simultaneously displayed views and the optimal angle between views can be different. In order to guide a player for selection of output views, alternative groups of output views and the common and differentiating characteristics between them can be indicated with the Multiview Group Relation box ('swtc'), which also includes the Multiview Relation Attributes box ('mvra').

When an MVC or MVD bitstream is represented by multiple tracks and a player uses an operating point that contains data in multiple tracks, the player must reconstruct MVC or MVD access units before passing them to the MVC or MVD decoder. An MVC or MVD operating point may be explicitly represented by a track, i.e., an access unit is reconstructed simply by resolving all extractors and aggregators of a sample. If the number of operating points is large, it may be space-consuming and impractical to create a track for each operating point. In such a case, MVC or MVD access units are reconstructed as specified in subclause 7.6.2. The MVC or MVD Decoder Configuration record contains a field indicating whether the associated samples use explicit or implicit access unit reconstruction (see the explicit_au_track field).

7.2   Overview of MVC or MVD storage

The storage of MVC and MVD streams can be supported by a number of structures, including information in the sample entry, the media information box, and sample groups. Table 5 provides an overview of the structures provided, their names, and a brief description of their functions.

NOTE   Each group of rows starting with an entry in the left column (e.g. 'minf', '?vc?') document a containment structure within that container; the higher level containment is not shown.

Table 5 — Box, sample entry and group structures for MVC and MVD streams

				Box name	Brief description
minf				Media Information Box	
	mvci			Multiview Information Box	
		mvcg		Multiview Group Box	Specifies a multiview group for the views of the multiview video stream that are output
			buff	Buffering Information Box	Contains the buffering information of the bitstream subset specified by the multiview group
			mvra	Multiview Relation Attribute Box	Indicates the relation of the tracks or tiers of the respective multiview group with each other (when contained in a Multiview Group box)
			tibr	Tier Bit Rate Box	Provides information about the bit rate values of the bitstream subset specified by the multiview group

Table 5 *(continued)*

				Box name	Brief description
			tiri	Tier Information Box	Provides information about the profile, level, frame size, discardability, and frame-rate of the bitstream subset specified by the multiview group
			vwdi	Multiview Scene Information Box	Indicates the maximum disparity in a scene with multiple views
		swtc		Multiview Group Relation Box	Specifies a set of multiview groups from which one multiview group is decoded and played at any time
			mvra	Multiview Relation Attribute Box	Indicates the relation of the multiview groups with each other (when contained in a Multiview Group Relation box)
?vc?				Sample Entry	(Note: various codes are used for sample entries)
		vsib		View Scalability Information SEI Message Box	Contains an SEI NAL unit containing only a view scalability information SEI message as specified in ISO/IEC 14496-10:2014 Annex H
		ecam		Extrinsic Camera Parameters Box	Contains camera parameters that define the location and orientation of the camera reference frame with respect to a known world reference frame
		icam		Intrinsic Camera Parameters Box	Contains camera parameters that link the pixel coordinates of an image point with the corresponding coordinates in the camera reference frame
		vwid		View Identifier Box	Indicates the views included in the track (when included in a sample entry)
		mvcP		MVC View Priority Assignment Box	Provides a URI containing a unique name of the method used to assign content_priority_id values for the View Priority sample grouping
		mvcC		MVC Configuration Box	
		mvdC		MVCD Configuration Box	Contains the MVD decoder configuration record and the MVD depth resolution box (for MVD streams only)
		3dpr		MVD Depth Resolution Box	Provides the resolution of depth views (for MVD streams only)
		3sib		MVD Scalability Information SEI Message Box	Contains an SEI NAL unit containing only an MVCD view scalability information SEI message as specified in ISO/IEC 14496-10:2014 Annex I
sgpd				Sample Group Description Box	
	mvif			Multiview Group Entry	Contains the following boxes
		buff		Buffering Information Box	Contains the buffer information of the tier
		ldep		Tier Dependency Box	Identifies the tiers that the current tier is dependent on
		svip		Initial Parameter Sets Box	Contains parameter sets needed for decoding this tier and all the tiers it depends on

Table 5 *(continued)*

				Box name	Brief description
		svpr		Priority Range Box	Reports the minimum and maximum priority_id of the NAL units mapped to this tier
		tibr		Tier Bit Rate Box	Provides information about the bit rate values of a tier
		tiri		Tier Information Box	Provides information about the profile, level, frame size, discardability, and frame-rate of a tier
		vipr		View Priority Box	Labels views with priorities based on content
		vwid		View Identifier Box	Indicates the views included in the tier (when included in a Multiview Group entry,)
	dtrt			Decode Re-timing Group Entry	Provides adjusted decoding times when high temporal layers are discarded
	scnm			Sample Map Group Entry	Provides the mapping of NAL units to multiview groups for all samples in the track

The structures within a sample entry provide information for the decoding or use of the samples (video information) that are associated with that sample entry. Sample groups provide time-varying information about the track as a whole, assisting (for example) with the extraction of subsets of the media within a track. Information in the Multiview Information Box (appearing in the media information box) can span several tracks and is descriptive of collections of tracks, even though the Multiview Information Box resides in the track containing the base view of the stream.

7.3   MVC and MVD elementary stream structures

MVC and MVD streams are stored in accordance with subclause 5.2, with the following definition of an MVC or MVD video elementary stream:

— **An MVC and MVD Video Elementary Stream** contains all video coding related NAL units (i.e. those NAL units containing video data or signalling video structure, possibly after resolution of extractors and aggregators) and may contain non-video coding related NAL units such as SEI messages and access unit delimiter NAL units. Aggregators and Extractors, when present, shall not directly output by file parsers.

MVC and MVD streams may also be stored using associated parameter set streams, when needed.

For MVC and MVD streams, Table 2 is amended by Table 6.

Table 6 — NAL unit types in MVC, MVD, and AVC streams

Value of nal_unit_type	Description	AVC video elementary stream	MVC video elementary stream (sample entry 'avc1', 'avc2', 'mvc1', or 'mvc2')	MVC video elementary stream (sample entry 'avc3', 'avc4', 'mvc3', or 'mvc4')	MVD video elementary stream (sample entry 'mvd1', 'mvd2', 'a3d1', or 'a3d2')	MVD video elementary stream (sample entry 'mvd3', 'mvd4', 'a3d3', or 'a3d3')	Parameter set elementary stream
14	Prefix NAL unit prefix_nal_unit_rbsp()	Not specified	Yes	Yes	Yes	Yes	No
15	Subset sequence parameter set subset_seq_parameter_set_rbsp()	Not specified	No	Yes Parameter set elementary stream shall not be used	No	Yes Parameter set elementary stream shall not be used	Yes
20	Coded slice extension slice_layer_extension_rbsp()	Not specified	Yes	Yes	Yes	Yes	No
21	Coded slice extension for a depth view component or a 3D-AVC texture view component slice_layer_extension_rbsp()	Not specified	Not specified	Not specified	Yes	Yes	No
24 – 31	Not specified	Not specified	See Annex F	See Annex F	See Annex F	See Annex F	See Annex F

There may be AVC VCL NAL units, MVC VCL NAL units and other NAL units, i.e. non-VCL NAL units, present in an MVC video elementary stream. There may be AVC VCL NAL units, MVC VCL NAL units, MVD VCL NAL units, and non-VCL NAL units present in an MVD video elementary stream. Additionally, there may be Aggregators or Extractors present in an MVC or MVD video elementary stream.

An AVC VCL NAL unit in an MVC or MVD video elementary stream conforming to one or more profiles specified in ISO/IEC 14496-10:2014, Annexes H, I and J shall be immediately preceded by a prefix NAL unit. In this document, an AVC VCL NAL unit and the immediately preceding prefix NAL unit are logically seen as one NAL unit.

7.4   Use of the plain AVC file format

The MVC or MVD file format is an extension of the plain AVC file format defined in subclause 4.11.

Subclause 5.4.7 is defined for use with plain AVC streams. Its use with MVC and MVD streams is deprecated.

7.5   Sample and configuration definition

7.5.1   Overview

MVC Sample: An MVC sample consists of one or more view components as defined in ISO/IEC 14496-10:2014, Annex H and the associated non-VCL NAL units.

MVD Sample: An MVD sample consists of one or more view components as defined in ISO/IEC 14496-10:2014, Annexes I and J and the associated non-VCL NAL units, where each view component may contain a texture view component, a depth view component or both.

7.5.2   Canonical order and restriction

The following restrictions apply to MVC and MVD data in addition to the requirements in subclause 5.3.2.

— **MVC coded slice NAL units** (Coded slice extension): All MVC coded slice NAL units for a single instant in time shall be contained in the sample whose composition time is that of the picture represented by the access unit. An MVC sample shall contain at least one AVC or MVC VCL NAL unit.

— **MVD VCL NAL units** (Coded slice extension): All MVD VCL NAL units for a single instant in time shall be contained in the sample whose composition time is that of the picture represented by the access unit. An MVD sample shall contain at least one AVC, MVC or MVD VCL NAL unit.

— **Prefix NAL units**: Each prefix NAL unit is placed immediately before the corresponding AVC VCL NAL unit.

— **Aggregators/Extractors**: The order of all NAL units included in an Aggregator or referenced by an Extractor is exactly the decoding order as if these NAL units were present in a sample not containing aggregators or extractors. After processing the Aggregator or the Extractor, all NAL units must be in valid decoding order as specified in ISO/IEC 14496-10.

7.5.3   Decoder configuration record

7.5.3.1   MVC decoder configuration record

When the AVC decoder configuration record (as defined in subclause 5.3.3.1) is used for a stream that can be interpreted as either an MVC or AVC stream, the AVC decoder configuration record shall reflect the properties of the AVC compatible base view, e.g. it shall contain only parameter sets needed for decoding the AVC base view.

If the sample entry name is 'mvc1' or 'mvc2', a parameter set stream may be used with MVC streams, as with AVC streams. In that case, parameter sets shall not be included in the decoder configuration

record. Otherwise (the sample entry name is `'mvc3'` or `'mvc4'`), parameter sets may be stored in both the decoder configuration record or as part of samples while a parameter set elementary stream shall not be used.

Sequence or picture parameter sets, including subset sequence parameter sets, are numbered in order of storage from 1 to `numOfSequenceParameterSets` or `numOfPictureParameterSets`, respectively. Sequence and picture parameter sets stored in this record in a file may be referenced using this 1-based index by the `InitialParameterSetBox`.

The `MVCDecoderConfigurationRecord` is structurally identical to an `AVCDecoderConfigurationRecord`. However, the reserved bits preceding and succeeding the `lengthSizeMinusOne` field are re-defined. The syntax is as follows:

```
aligned(8) class MVCDecoderConfigurationRecord {

   unsigned int(8) configurationVersion = 1;

   unsigned int(8) AVCProfileIndication;

   unsigned int(8) profile_compatibility;

   unsigned int(8) AVCLevelIndication;

   unsigned int(1) complete_representation;

   unsigned int(1) explicit_au_track;

   bit(4) reserved = '1111'b;

   unsigned int(2) lengthSizeMinusOne;

   bit(1) reserved = '0'b;

   unsigned int(7) 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;

   }

}
```

The semantics of the fields `AVCProfileIndication`, `profile_compatibility`, and `AVCLevelIndication` differ from the `AVCDecoderConfigurationRecord` as follows:

The fields `AVCProfileIndication`, `AVCLevelIndication` carry the profile and level indications, respectively, indicating the profile and level for the bitstream represented by this track, i.e., the bitstream that contains all the views of this track and the views required for decoding of this track and wherein all the views in this track are the target output views. If `AVCLevelIndication` is equal to 0, the level that applies to the bitstream defined above operating with all the views of this track being the target output views is unspecified. `AVCProfileIndication`, `profile_compatibility`, and

AVCLevelIndication, if non-zero, shall have values such that an MVC decoder supporting these values is able to decode bitstreams conforming to the profile, level and profile compatibility flags indicated in any of the sequence parameter sets or subset sequence parameter sets contained in this record.

The semantics of other fields are as follows, or, if not present in the following, are as defined for an AVCDecoderConfigurationRecord:

— complete_representation is set on a minimal set of tracks that contain a portion of the original encoded stream, as defined in subclause 7.6.1. Other tracks may be removed from the file without loss of any portion of the original encoded bitstream, and, once the set of tracks has been reduced to only those in the complete subset, any further removal of a track removes a portion of the encoded information.

— explicit_au_track is set on a track that is 'complete'; it is not necessary to determine the view dependencies, nor calculate whether views not present in this track should be found from other tracks. However, subject to the rules for the sample entry types, extractors may be present and need to be followed to gather all the NAL units needed.

— numOfSequenceParameterSets indicates the number of SPSs and subset SPSs that are used for decoding the MVC elementary stream.

— SequenceParameterSetLength indicates the length in bytes of the SPS or subset SPS NAL unit.

— SequenceParameterSetNALUnit contains a SPS or subset SPS NAL unit as specified in ISO/IEC 14496-10:2014, Annex H. SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Subset SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Any SPS shall occur before all the subset SPSs, if any.

7.5.3.2 MVD decoder configuration record

The syntax structure of MVDDecoderConfigurationRecord is exactly the same as MVCDecoderConfigurationRecord.

When the AVC decoder configuration record (as defined in subclause 5.3.3.1) is used for a stream that can be interpreted as an MVD stream, the AVC decoder configuration record shall reflect the properties of the AVC compatible base view, e.g. it may contain only parameter sets needed for decoding the AVC base view.

When the MVC decoder configuration record (as defined in subclause 7.5.3.1) is used for a stream that can be interpreted as an MVC or MVD stream, the MVC decoder configuration record shall reflect the properties of the MVC compatible bitstream subset (i.e. the bitstream subset with only the texture views), e.g. it may contain only parameter sets needed for decoding the MVC compatible bitstream subset.

If the sample entry name is 'mvd1', 'mvd2', 'a3d1', or 'a3d2', a parameter set stream may be used with MVD streams, as with AVC and MVC streams. In that case, parameter sets shall not be included in the decoder configuration record. Otherwise (the sample entry name is 'mvd3', 'mvd4', 'a3d3' or 'a3d4'), parameter sets may be stored in both the decoder configuration record or as part of samples while a parameter set elementary stream shall not be used.

Sequence or picture parameter sets, including subset sequence parameter sets, are numbered in order of storage from 1 to numOfSequenceParameterSets or numOfPictureParameterSets, respectively. Sequence and picture parameter sets stored in this record in a file may be referenced using this 1-based index by the InitialParameterSetBox.

The semantics of the fields AVCProfileIndication, profile_compatibility, and AVCLevelIndication differ from the MVCDecoderConfigurationRecord as follows. AVCProfileIndication, profile_compatibility, and AVCLevelIndication, if non-zero, shall have values such that a MVD decoder supporting these values is able to decode bitstreams conforming to the profile, level and profile compatibility flags indicated in any of the sequence parameter sets or subset sequence parameter sets contained in this record.

The semantics of other fields are as follows, or, if not present, are as defined for an MVCDecoderConfigurationRecord:

— `numOfSequenceParameterSets` indicates the number of SPSs and subset SPSs that are used for decoding the MVD elementary stream.

— `sequenceParameterSetNALUnit`, when contained in the MVCD Configuration Box, contains a SPS or subset SPS NAL unit as specified in ISO/IEC 14496-10:2014, Annex I. `sequenceParameterSetNALUnit`, when contained in the A3D Configuration Box, contains a SPS or subset SPS NAL unit as specified in ISO/IEC 14496-10:2014, Annex J. SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Subset SPSs shall occur in order of ascending parameter set identifier with gaps being allowed. Any SPS shall occur before all the subset SPSs, if any.

7.6 Derivation from the ISO base media file format

7.6.1 MVC and MVD track structures

An MVC or MVD stream is represented by one or more video tracks in a file. Each track represents one or more views of the stream. For a track in a file storing an MVD video stream, the track may contain texture only, depth only or both texture and depth.

There is a minimal set of one or more tracks that, when taken together, contain the complete set of encoded information. All these tracks shall have the flag "`complete_representation`" set in all their sample entries. This group of tracks that form the complete encoded information are called the "complete subset".

The track that has the flag "`complete_representation`" set and contains NAL units of the base view with temporal_id equal to 0 shall be nominated as the 'base view track'. All the other tracks that are part of the same stream shall be linked to this base track by means of a track reference of type `'sbas'` (view base). The complete encoded information can be retained when the tracks included in the "complete subset" are retained; all other tracks shall represent subsets, copies (i.e. the same media data) or re-orderings of the complete subset.

All the tracks sharing the same base view track shall share the same timescale as the scalable base track. For MVD streams, all the tracks containing the texture view and the depth view of a particular view shall share the same timescale. Note that the texture view and the depth view of a particular view have the same value of view_id.

If a view represented by a track uses another view represented by another track as an inter-view prediction reference, a track reference of type `'scal'` shall be included in the track referring to the source track for inter-view prediction.

For MVD streams, if a depth view is stored in a different track than the track containing the texture view associated with the depth view, a track reference of type `'deps'` shall be included in the track containing the depth view, referring to the track containing the texture view. The presence of this track reference indicates that the current track contains the depth view that is associated with a texture view in the referenced track.

NOTE   If a track containing a part of an MVC or MVD bitstream is removed from a file, care should be taken to remove also those tracks that contain 'scal' and 'sbas' track references to the removed track and references to the multiview groups that include the removed track.

7.6.2 Reconstruction of an access unit

In order to reconstruct an access unit from samples of one or more MVC or MVD tracks, the target output views may need to be determined first, by examining the Multiview Group box (7.7.3) and the Multiview Group Relation box (7.7.4). The `explicit_au_track` flag equal to 1 states that this track is a complete operating point; nonetheless, the track should be examined to determine which views delivered by this track are the output views.

If the target output views are not exactly represented by any track marked with `explicit_au_track` equal to 1 in the MVC decoder configuration record, access units are reconstructed as follows.

The views that are required for decoding the determined target output views can be concluded from reference view identifiers included in the View Identifier box, the `'scal'` track references, or Tier Dependency boxes.

If several tracks contain data for the access unit, the alignment of respective samples in tracks is performed on decoding time, i.e. using the time-to-sample table only without considering edit lists.

An access unit is reconstructed from the respective samples in the required tracks and tiers by arranging their NAL units in an order conforming to ISO/IEC 14496-10. Duplicate copies of NAL units, when present, need to be removed in order to conform to ISO/IEC 14496-10. The following order provides an outline of the procedure to form a conforming access unit:

— All parameter set NAL units (from the associated parameter set tracks and from the associated elementary stream tracks).

— All SEI NAL units (from the associated parameter set tracks and from the associated elementary stream tracks).

— View components in ascending order of view order index value.

— Within a view component, if both texture and depth are present, then the texture view component precedes the depth view component.

— NAL units within a texture or depth view component are in their appearance order within the sample.

7.6.3 Sample entry

7.6.3.1 Boxes for sample entry

7.6.3.1.1 Intrinsic camera parameters box

7.6.3.1.1.1 Definition

Box Type: `'icam'`

Container: Sample Entry (`'avc1'`, `'avc2'`, `'avc3'`, `'avc4'`, `'mvc1'`, `'mvc2'`, `'mvc3'`, `'mvc4'`, `'mvd1'`, `'mvd2'`, `'mvd3'`, `'mvd4'`, `'a3d1'`, `'a3d2'`, `'a3d3'`, `'a3d4'`)

Mandatory: No

Quantity: Zero or more

This subclause specifies intrinsic camera parameters that link the pixel coordinates of an image point with the corresponding coordinates in the camera reference frame. A specification of focal length and parameters related to the geometric distortion due to camera optics is given in ISO/IEC 14496-10:2014, Annex H.

7.6.3.1.1.2 Syntax

```
class IntrinsicCameraParametersBox extends FullBox ('icam', version=0, flags) {
    unsigned int(6)    reserved=0;
    unsigned int(10)   ref_view_id;
    unsigned int(32)   prec_focal_length;
    unsigned int(32)   prec_principal_point;
    unsigned int(32)   prec_skew_factor;
    unsigned int(8)    exponent_focal_length_x;
    signed   int(64)   mantissa_focal_length_x;
    unsigned int(8)    exponent_focal_length_y;
    signed   int(64)   mantissa_focal_length_y;
    unsigned int(8)    exponent_principal_point_x;
    signed   int(64)   mantissa_principal_point_x;
    unsigned int(8)    exponent_principal_point_y;
    signed   int(64)   mantissa_principal_point_y;
    unsigned int(8)    exponent_skew_factor;
    signed   int(64)   mantissa_skew_factor;
}
```

7.6.3.1.1.3 Semantics

reserved this field shall be equal to zero.

ref_view_id indicates the view_id identifying a view for which intrinsic camera parameters are indicated in this Intrinsic Camera Parameters Box.

prec_focal_length specifies the exponent of the maximum allowable truncation error for focal_length_x and focal_length_y as given by $2^{-prec\_focal\_length}$. The value of prec_focal_length shall be in the range of 0 to 31, inclusive.

prec_principal_point specifies the exponent of the maximum allowable truncation error for principal_point_x and principal_point_y as given by $2^{-prec\_principal\_point}$. The value of prec_principal_point shall be in the range of 0 to 31, inclusive.

prec_skew_factor specifies the exponent of the maximum allowable truncation error for skew factor as given by $2^{prec\_skew\_factor}$. The value of prec_skew_factor shall be in the range of 0 to 31, inclusive.

exponent_focal_length_x specifies the exponent part of the focal length in the horizontal direction. The value of exponent_focal_length_x shall be in the range of 0 to 62, inclusive. The value 63 is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified focal length.

mantissa_focal_length_x specifies the mantissa part of the focal length of the i-th camera in the horizontal direction.

exponent_focal_length_y specifies the exponent part of the focal length in the vertical direction. The value of exponent_focal_length_y shall be in the range of 0 to 62, inclusive. The value 63 is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified focal length.

mantissa_focal_length_y specifies the mantissa part of the focal length in the vertical direction.

mantissa_principal_point_x specifies the mantissa part of the principal point in the horizontal direction.

exponent_principal_point_y specifies the exponent part of the principal point in the vertical direction. The value of exponent_principal_point_y shall be in the range of 0 to 62, inclusive. The value 63 is

reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified principal point.

`mantissa_principal_point_y` specifies the mantissa part of the principal point in the vertical direction.

`exponent_skew_factor` specifies the exponent part of the skew factor. The value of `exponent_skew_factor` shall be in the range of 0 to 62, inclusive. The value 63 is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified skew factor.

`mantissa_skew_factor` specifies the mantissa part of the skew factor.

The intrinsic matrix A for the camera associated to the view indicated by `ref_view_id` is represented as follows:

$$\begin{bmatrix} focalLengthX & skewFactor & principalPointX \\ 0 & focalLengthY & principalPointY \\ 0 & 0 & 1 \end{bmatrix}$$

Each component of the intrinsic matrix is obtained from the variables specified in Table 7 as the variable x computed as follows.

If $0 < e < 63$, $x = 2^{e-31} * (1 + n \div 2v)$, with $v = \max(0, e + p - 31)$ (1)

If e is equal to 0, $x = 2^{-(30+v)} * n$, with $v = \max(0, p - 30)$ (2)

Table 7 — Association between camera parameter variables and syntax elements

x	e	n	p
focalLengthX	exponent_focal_length_x	mantissa_focal_length_x	prec_focal_length
focalLengthY	exponent_focal_length_y	mantissa_focal_length_y	prec_focal_length
principalPointX	exponent_principal_point_x	mantissa_principal_point_x	prec_principal_point
principalPointY	exponent_principal_point_y	mantissa_principal_point_y	prec_principal_point
skewFactor	exponent_skew_factor	mantissa_skew_factor	prec_skew_factor

7.6.3.1.2   Extrinsic camera parameters box

7.6.3.1.2.1   Definition

Box Type: `'ecam'`

Container: Sample Entry (`'avc1'`, `'avc2'`, `'avc3'`, `'avc4'`, `'mvc1'`, `'mvc2'`, `'mvc3'`, `'mvc4'`, `'mvd1'`, `'mvd2'`, `'mvd3'`, `'mvd4'`, `'a3d1'`, `'a3d2'`, `'a3d3'`, `'a3d4'`)

Mandatory: No

Quantity: Zero or more

This subclause specifies extrinsic camera parameters that define the location and orientation of the camera reference frame with respect to a known world reference frame. A specification of extrinsic camera parameters including translation vector and rotation matrix is given in ISO/IEC 14496-10:2014, Annex H.

The extrinsic camera parameters are specified according to a right-handed coordinate system, where the upper left corner of the image is the origin, i.e., the (0, 0) coordinate, with the other corners of the image having non-negative coordinates. With these specifications, a 3-dimensional world point, wP=[x y z] is mapped to a 2-dimensional camera point, cP = [u v 1], according to:

$$s * cP = A * R^{-1} * (wP - T)$$

where A denotes the intrinsic camera parameter matrix that can be indicated by an intrinsic camera parameters box (see subclause 7.6.3.1.1), R^{-1} denotes the inverse of the rotation matrix R, T denotes the translation vector, and s (a scalar value) is an arbitrary scale factor chosen to make the third coordinate of cP equal to 1. The elements of A, R, T are determined according the syntax elements signalled in this box and as specified below.

7.6.3.1.2.2 Syntax

```
class ExtrinsicCameraParametersBox extends FullBox ('ecam', version=0, flags) {
    unsigned int(6)    reserved=0;
    unsigned int(10)   ref_view_id;
    unsigned int(8)    prec_rotation_param;
    unsigned int(8)    prec_translation_param;
    for (j=1; j<=3; j++) { /* row */
       for (k=1; k<=3; k++) { /* column */
          unsigned int(8)    exponent_r[j][k];
          signed   int(64)   mantissa_r [j][k];
       }
       unsigned int(8)     exponent_t[j];
       signed   int(64)  mantissa_t[j];
    }
}
```

7.6.3.1.2.3 Semantics

`reserved` this field shall be equal to zero.

`ref_view_id` indicates the view_id identifying a view for which intrinsic camera parameters are indicated in this Intrinsic Camera Parameters Box.

`prec_rotation_param` specifies the exponent of the maximum allowable truncation error for r[j][k] as given by $2^{prec\_rotation\_param}$. The value of `prec_rotation_param` shall be in the range of 0 to 31, inclusive.

`prec_translation_param` specifies the exponent of the maximum allowable truncation error for t[j] as given by $2^{prec\_translation\_param}$. The value of `prec_ translation_param` shall be in the range of 0 to 31, inclusive.

`exponent_r[j][k]` specifies the exponent part of (j, k) component of the rotation matrix. The value of `exponent_r[j][k]` shall be in the range of 0 to 62, inclusive. The value 63 is reserved for future use by ITU-T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified rotation matrix.

`mantissa_r[j][k]` specifies the mantissa part of (j, k) component of the rotation matrix.

`exponent_t[j]` specifies the exponent part of the j-th component of the translation vector. The value of `exponent_t[j]` shall be in the range of 0 to 62, inclusive. The value 63 is reserved for future use by ITU T | ISO/IEC. Decoders shall treat the value 63 as indicating an unspecified translation vector.

`mantissa_t[j]` specifies the mantissa part of the j-th component of the translation vector.

The rotation matrix R is represented as follows:

$$\begin{bmatrix} rE[0][0] & rE[0][1] & rE[0][2] \\ rE[1][0] & rE[1][1] & rE[1][2] \\ rE[2][0] & rE[2][1] & rE[2][2] \end{bmatrix}$$

The translation vector T is represented as follows:

$$\begin{bmatrix} tE[0] \\ tE[1] \\ tE[2] \end{bmatrix}$$

Each component of the rotation matrix and the translation vector is obtained from the variables specified in Table 8 as the variable x computed as follows.

If $0 < e < 63$, $x = 2^{e-31} * (1 + n \div 2^v)$, with $v = \max(0, e + p - 31)$ (3)

If e is equal to 0, $x = 2^{-(30+v)} * n$, with $v = \max(0, p - 30)$ (4)

Table 8 — Association between camera parameter variables and syntax elements

x	e	n	p
rE[j][k]	exponent_r[j][k]	mantissa_r[j][k]	prec_rotation_param
tE[j]	exponent_t[j]	mantissa_t[j]	prec_translation_param

7.6.3.1.3   View identifier box

7.6.3.1.3.1   Definition

Box Type: 'vwid'

Container: Sample Entry ('avc1', 'avc2', 'avc3', 'avc4',
                'mvc1', 'mvc2', 'mvc3', 'mvc4',
                'mvd1', 'mvd2', 'mvd3', 'mvd4',
                'a3d1', 'a3d2', 'a3d3', 'a3d4')

                or MultiviewGroupEntry

Mandatory: Yes (for sample entries and the primary group definition in Multiview Group entries)

Quantity: Exactly one (for sample entries and the primary group definition in Multiview Group entries)

                Zero for non-primary group definitions in Multiview Group entries

When included in a sample entry, this box indicates the views included in the track. When included in a Multiview Group entry, this box indicates the views included in the respective tier. This box also indicates the view order index for each listed view. Additionally, the box includes the minimum and maximum values of temporal_id included in the track or tier when the View Identifier box is included in a sample entry or Multiview Group entry, respectively. Moreover, the box indicates the referenced views required for decoding the views included in the track or tier. Moreover, for MVD streams, the box indicates, for each of the view included in the track, the presence of texture and/or depth in the track and in the stream.

7.6.3.1.3.2 Syntax

```
class ViewIdentifierBox extends FullBox ('vwid', version=0, flags)
{
    unsigned int(2)   reserved6 = 0;
    unsigned int(3)   min_temporal_id;
    unsigned int(3)   max_temporal_id;
    unsigned int(16)  num_views;
    for (i=0; i<num_views; i++) {
        unsigned int(6)   reserved1 = 0;
        unsigned int(10)  view_id[i];
        unsigned int(6)   reserved2 = 0;
        unsigned int(10)  view_order_index;
        unsigned int(1)   texture_in_stream[i];
        unsigned int(1)   texture_in_track[i];
        unsigned int(1)   depth_in_stream[i];
        unsigned int(1)   depth_in_track[i];
        unsigned int(2)   base_view_type;
        unsigned int(10)  num_ref_views;
        for (j = 0; j < num_ref_views; j++) {
            unsigned int(4)   reserved5 = 0;
            unsigned int(2)   dependent_component_idc[i][j];
            unsigned int(10)  ref_view_id[i][j];
        }
    }
}
```

7.6.3.1.3.3 Semantics

min_temporal_id, max_temporal_id take the minimum and maximum value, respectively, of the temporal_id syntax element that is present in the NAL unit header extension of the NAL units mapped to the track or tier when the View Identifier box is included in a sample entry or Multiview Group entry, respectively. For AVC streams this takes the value that is, or would be, in the prefix NAL unit.

num_views, when the View Identifer box is present in a sample entry, indicates the number of views included in the track. When the View Identifier box is present in a Multiview Group entry, num_views indicates the number of views included in the respective tier.

view_id[i] indicates the value of the view_id syntax element in the NAL unit header extension of a view included in the track or tier when the View Identifier box is included in a sample entry or Multiview Group entry, respectively.

view_order_index indicates the value of the VOIdx variable, as specified in ISO/IEC 14496-10:2014, Annex H, for a view included in the track or tier when the View Identifier box is included in a sample entry or Multiview Group entry, respectively.

texture_in_stream[i] equal to 1 indicates that the texture view for the view with view_id[i] is present in the stream. The value indicates that the texture view for the view with view_id[i] is not present in the stream.

texture_in_track[i] equal to 1 indicates that the texture view for the view with view_id[i] is present in the track. The value indicates that the texture view for the view with view_id[i] is not present in the track. When texture_in_stream[i] is equal to 0, the value of texture_in_track[i] shall be equal to 0.

depth_in_stream[i] equal to 1 indicates that the depth view for the view with view_id[i] is present in the stream. The value indicates that the depth view for the view with view_id[i] is not present in the stream. When texture_in_stream[i] is equal to 0, the value of depth_in_stream[i] shall be equal to 1.

depth_in_track[i] equal to 1 indicates that the depth view for the view with view_id[i] is present in the track. The value indicates that the depth view for the view with view_id[i] is not present in the track. When depth_in_stream[i] is equal to 0, the value of depth_in_track[i] shall be equal to 0. When texture_in_track[i] is equal to 0, the value of depth_in_track[i] shall be equal to 1.

base_view_type indicates whether the view is a base view (virtual or not). It takes the following values:

— 0 indicates that the view is neither a base view nor virtual base view.

— 1 shall be used to label the non-virtual base view of the MVC bitstream.

— 2 is a reserved value and shall not be used.

— 3 indicates that the view with view_id[i] is a virtual base view. The respective independently coded non-base view with view_id[i] resides in another track. When base_view_type is equal to 3, the subsequent num_ref_views shall be equal to 0.

num_ref_views indicates the number of views that may be directly or indirectly referenced by the view with view_id[i].

dependent_component_idc[i][j] indicates how the texture view and depth view of the j-th reference view are required for decoding the view with view_id[i]. If the value is equal to 0, only the texture view of the reference view is required. If the value is equal to 1, only the depth view of the reference view is required. If the value is equal to 2, both texture view and depth view of the reference view are required. The value of 3 is reserved.

ref_view_id[i][j] indicates the view identifier of the j-th view that may be directly or indirectly referenced by the view with view_id[i], i.e., that may be required for decoding of the view with view_id[i]. If a view is required for decoding the view with view_id[i], it shall be listed as one of ref_view_id[i][j]. When the View Identifier box is included in a sample entry, it is recommended to indicate the referenced views for both anchor and non-anchor access units in the same sample entry.

7.6.3.2 MVC and MVD sample entry definitions

7.6.3.2.1 Definition

Sample Entry and Box Types:

```
'mvc1', 'mvc2', 'mvc3', 'mvc4',
'mvd1', 'mvd2', 'mvd3', 'mvd4',
'a3d1', 'a3d2', 'a3d3', 'a3d4'
```

Container: Sample Description Box ('stsd')

Mandatory: One of the above listed sample entries is mandatory

Quantity: One or more sample entries may be present

When present, the AVC Configuration Box documents the Profile, Level, and possibly also parameter sets pertaining to the AVC compatible base view as defined by the AVCDecoderConfigurationRecord. When present, the MVC Configuration Box documents the Profile, Level and Parameter Set information pertaining to the entire MVC stream as defined by the MVCDecoderConfigurationRecord. When present, the MVCD Configuration Box documents the Profile, Level and Parameter Set information pertaining to the entire MVC+D stream as defined by the MVDDecoderConfigurationRecord. When present, the A3D Configuration Box documents the Profile, Level and Parameter Set information pertaining to the entire 3D-AVC stream as defined by the MVDDecoderConfigurationRecord.

For the AVC sample entries 'avc1', 'avc2', 'avc3' and 'avc4', the width and height fields in the sample entry document the cropped frame dimensions of the AVC base layer. For the MVC sample entries 'mvc1', 'mvc2', 'mvc3', and 'mvc4', and for the MVD sample entries 'mvd1', 'mvd2', 'mvd3', 'mvd4', 'a3d1', 'a3d2', 'a3d3', and 'a3d4', the width and height fields in the sample entry document the cropped frame dimensions achieved by decoding any single texture view of the entire MVC or MVD stream. Futhermore, for MVD sample entries 'mvd1', 'mvd2', 'mvd3', 'mvd4', 'a3d1', 'a3d2', 'a3d3', and 'a3d4', the depth_width and depth_height in the MVDDepthResolutionBox document the cropped frame dimensions achieved by decoding any single depth view of the entire MVD stream.

The lengthSizeMinusOne field in the AVC, MVC, MVCD, and A3D configurations in any given sample entry shall have the same value.

A priority assignment URI provides the name (in the URI space) of a method used to assign priority_id values. When it occurs in an AVC or MVC sample entry, exactly one URI shall be present, that documents the priority_id assignments in the entire AVC or MVC stream. An MVCD or A3D sample entry shall not contain an MVCViewPriorityAssignmentBox (and consequently priority assignment URI can be used neither for MVC+D streams nor for 3D-AVC streams).

The URI is treated here as a name only; it should be de-referenceable, though this is not required. File readers may be able to recognize some methods and thereby know what stream extraction operations based on priority_id would do.

The requirements for the sample entry types 'avc1' and 'avc2' as documented in subclause 6.5.3.1.1 also apply here.

When present in an AVC, MVC, MVCD, or A3D sample entry, ViewScalabilityInfoSEIBox, ViewIdentifierBox, IntrinsicCameraParametersBox, ExtrinsicCameraParametersBox, MVDScalabilityInformationSEIBox, BitRateBox and MPEG4ExtensionDescriptorsBox, apply to the entire AVC, MVC, MVC+D, or 3D-AVC stream, respectively.

The parameter sets required to decode a NAL unit that is present in the sample data of a video stream, either directly or by reference from an Extractor, shall be present in the decoder configuration of that video stream or in the associated parameter set stream (if used).

Table 9 shows for a video track all the possible uses of sample entries, configurations, and the MVC tools (excluding timed metadata, which is always used in another track).

Table 9 — Use of sample entries for AVC, MVC, MVC+D and 3D-AVC tracks

Sample entry name	With configuration records	Meaning
`'avc1'` or `'avc3'`	AVC Configuration Only	A plain AVC track with AVC NAL units only; Extractors, aggregators, and tier grouping shall not be present.
`'avc2'` or `'avc4'`	AVC Configuration Only	A plain AVC track with AVC NAL units only; Extractors may be present; Aggregators may be present to contain and reference AVC NAL units; Tier grouping may be present.
`'avc1'` or `'avc3'`	AVC and MVC Configurations	An MVC track with both AVC and MVC NAL units; Aggregators and Extractors shall not be present.
`'avc2'` or `'avc4'`	AVC and MVC Configurations	An MVC track with both AVC NAL units and MVC NAL units; Extractors may be present and used to reference both AVC and MVC NAL units; Aggregators may be present to contain and reference both AVC and MVC NAL units; Tier grouping may be present.
`'mvc1'` or `'mvc3'`	MVC Configuration Only	An MVC track without AVC NAL units; Aggregators may be present to contain and reference MVC NAL units; Tier grouping may be present.
`'mvc2'` or `'mvc4'`	MVC Configuration Only	An MVC track without AVC NAL units; Extractors may be present and used to reference MVC NAL units; Aggregators may be present to contain and reference MVC NAL units; Tier grouping may be present.
`'avc1'` or `'avc3'`	AVC, MVC, and MVCD Configurations	An MVC+D track with AVC, MVC and MVC+D depth NAL units; Aggregators and extractors shall not be present; Tier grouping may be present.
`'mvc1'` or `'mvc3'`	MVC and MVCD Configurations	An MVC+D track without AVC NAL units but with MVC and MVC+D depth NAL units; Aggregators and tier grouping may be present.
`'mvc2'` or `'mvc4'`	MVC and MVCD Configurations	An MVC+D track without AVC NAL units but with MVC and MVC+D depth NAL units; Extractors, aggregators and tier grouping may be present.
`'mvd1'` or `'mvd3'`	MVCD Configuration Only	An MVC+D track with MVC+D depth NAL units only; Aggregators and tier grouping may be present.
`'mvd2'` or `'mvd4'`	MVCD Configuration Only	An MVC+D track with MVC+D depth NAL units only; Extractors, aggregators and tier grouping may be present.
`'avc1'` or `'avc3'`	AVC, MVC, MVCD, and A3D Configurations	A 3D-AVC track with AVC, MVC, MVC+D depth, and 3D-AVC NAL units; Aggregators and extractors shall not be present; Tier grouping may be present. If MVC Configuration were not present in this example, the track would not contain or refer to MVC NAL units. If MVCD Configuration were not present in this example, the track would not contain or refer to MVC+D depth NAL units.
`'mvc1'` or `'mvc3'`	MVC, MVCD, and A3D Configurations	A 3D-AVC track without AVC NAL units but with MVC, MVC+D depth NAL units, and 3D-AVC NAL units; Aggregators and tier grouping may be present. If MVCD Configuration were not present in this example, the track would not contain or refer to MVC+D depth NAL units.

Table 9 *(continued)*

Sample entry name	With configuration records	Meaning
`'mvc2'` or `'mvc4'`	MVC, MVCD, and A3D Configurations	A 3D-AVC track without AVC NAL units but with MVC, MVC+D depth NAL units, and 3D-AVC NAL units; Extractors, aggregators and tier grouping may be present. If MVCD Configuration were not present in this example, the track would not contain or refer to MVC+D depth NAL units.
`'mvd1'` or `'mvd3'`	MVCD and A3D Configurations	A 3D-AVC track without AVC or MVC NAL units but with MVC+D depth NAL units and 3D-AVC NAL units; Aggregators and tier grouping may be present.
`'mvd2'` or `'mvd4'`	MVCD and A3D Configurations	A 3D-AVC track without AVC or MVC NAL units but with MVC+D depth NAL units and 3D-AVC NAL units; Extractors, aggregators and tier grouping may be present.
`'a3d1'` or `'a3d3'`	A3D Configuration Only	A 3D-AVC track with 3D-AVC NAL units only; Aggregators and tier grouping may be present.
`'a3d2'` or `'a3d4'`	A3D Configuration Only	A 3D-AVC track with 3D-AVC NAL units only; Extractors, aggregators and tier grouping may be present.

7.6.3.2.2   Syntax

```
class MVCConfigurationBox extends Box('mvcC') {
    MVCDecoderConfigurationRecord() MVCConfig;
}
class ViewScalabilityInformationSEIBox extends Box('vsib', size)
{
    unsigned int(8*size-64) mvcscalinfosei;
}
class MVDDepthResolutionBox extends Box('3dpr')
{
    unsigned int(16) depth_width;
    unsigned int(16) depth_height;
/* The following 5 fields are collectively optional; they are either all present or all
absent. When grid_pos_num_views is not present, the for loop is not present, equivalent
to grid_pos_num_views equal to 0. These fields may be present or absent whenever the box
is present (e.g., in MVCDConfigurationBox or A3DConfigurationBox). */
    unsigned int(16) depth_hor_mult_minus1; // optional
    unsigned int(16) depth_ver_mult_minus1; // optional
    unsigned int(4) depth_hor_rsh; // optional
    unsigned int(4) depth_ver_rsh; // optional
    unsigned int(16) grid_pos_num_views; // optional
    for(i = 0; i < grid_pos_num_vies; i++) {
```

```
      bit(6) reserved=0;
      unsigned int(10) grid_pos_view_id[i];
      signed int(16) grid_pos_x[grid_pos_view_id[i]];
      signed int(16) grid_pos_y[grid_pos_view_id[i]];
   }
}
class MVCDConfigurationBox extends Box('mvdC') {
   MVDDecoderConfigurationRecord MVDConfig;
   MVDDepthResolutionBox mvdDepthRes;   //Optional
}
class A3DConfigurationBox extends Box('a3dC') {
   MVDDecoderConfigurationRecord MVDConfig;
   MVDDepthResolutionBox mvdDepthRes;   //Optional
}
class MVDScalabilityInformationSEIBox extends Box('3sib', size)
{
   unsigned int(8*size-64) mvdscalinfosei;
}
class AVCMVCSampleEntry() extends AVCSampleEntry ('avc1' or 'avc3') {
   ViewScalabilityInformationSEIBox scalability;  // optional
   ViewIdentifierBox    view_identifiers;  // optional
   MVCConfigurationBox  mvcconfig;   // optional
   MVCViewPriorityAssignmentBox  view_priority_method;   // optional
   IntrinsicCameraParametersBox  intrinsic_camera_params;    // optional
   ExtrinsicCameraParametersBox  extrinsic_camera_params;    // optional
   MVCDConfigurationBox mvcdconfig; // optional
   MVDScalabilityInformationSEIBox  mvdscalinfosei;   // optional
   A3DConfigurationBox  a3dconfig;   // optional
}
class AVC2MVCSampleEntry() extends AVC2SampleEntry ('avc2' or 'avc4') {
   ViewScalabilityInformationSEIBox scalability;  // optional
   ViewIdentifierBox    view_identifiers;  // optional
   MVCConfigurationBox  mvcconfig;   // optional  MVCViewPriorityAssignmentBox
   view_priority_method;//    optional
   IntrinsicCameraParametersBox  intrinsic_camera_params;   // optional
   ExtrinsicCameraParametersBox  extrinsic_camera_params // optional
   MVCDConfigurationBox mvcdconfig; // optional
   MVDScalabilityInformationSEIBox  mvdscalinfosei;   // optional
   A3DConfigurationBox  a3dconfig;  // optional
}
// Use this if the track is NOT AVC compatible
class MVCSampleEntry() extends VisualSampleEntry ('mvc1', 'mvc2',
                                                  'mvc3', or 'mvc4') {
```

```
    MVCConfigurationBox  mvcconfig;       // mandatory
    ViewScalabilityInformationSEIBox scalability;  // optional
    ViewIdentifierBox view_identifiers;      // mandatory
    MPEG4ExtensionDescriptorsBox descr;      // optional
    MVCViewPriorityAssignmentBox  view_priority_method;  // optional
    IntrinsicCameraParametersBox  intrinsic_camera_params;   // optional
    ExtrinsicCameraParametersBox  extrinsic_camera_params // optional
    MVCDConfigurationBox mvcdconfig; // optional
    MVDScalabilityInformationSEIBox  mvdscalinfosei;  // optional
    A3DConfigurationBox  a3dconfig;   // optional
}
 class MVCDSampleEntry() extends VisualSampleEntry ('mvd1', 'mvd2',
                                        'mvd3', or 'mvd4') {
    MVCDConfigurationBox mvcdconfig;  // mandatory
    MVDScalabilityInformationSEIBox  mvdscalinfosei;   // optional
    ViewIdentifierBox view_identifiers;      // mandatory
    MPEG4ExtensionDescriptorsBox descr;   // optional
    IntrinsicCameraParametersBox  intrinsic_camera_params;    // optional
    ExtrinsicCameraParametersBox  extrinsic_camera_params // optional
    A3DConfigurationBox  a3dconfig;   // optional
}
 class A3DSampleEntry() extends VisualSampleEntry ('a3d1', 'a3d2',
                                        'a3d3', or 'a3d4') {
    A3DConfigurationBox  a3dconfig;   // mandatory
    MVDScalabilityInformationSEIBox  mvdscalinfosei;  // optional
    ViewIdentifierBox view_identifiers;      // mandatory
    MPEG4ExtensionDescriptorsBox descr;   // optional
    IntrinsicCameraParametersBox  intrinsic_camera_params;   // optional
    ExtrinsicCameraParametersBox  extrinsic_camera_params // optional
}
```

7.6.3.2.3   Semantics

When the sample entry is `'mvc1'`, `'mvc2'`, `'mvc3'`, or `'mvc4'`, Compressorname in the base class VisualSampleEntry indicates the name of the compressor used, with the value "\012MVC Coding" being recommended (\012 is 10, the length of the string "MVC coding" in bytes).

When the sample entry is `'mvd1'`, `'mvd2'`, `'mvd3'`, or `'mvd4'`, Compressorname in the base class VisualSampleEntry indicates the name of the compressor used, with the value "\013MVCD Coding" being recommended (\013 is 11, the length of the string "MVCD coding" in bytes).

When the sample entry is `'a3d1'`, `'a3d2'`, `'a3d3'`, or `'a3d4'`, Compressorname in the base class VisualSampleEntry indicates the name of the compressor used, with the value "\012A3D Coding" being recommended (\012 is 10, the length of the string "A3D coding" in bytes).

depth_width and depth_height give the values of the width and height, respectively, of cropped frame dimensions of the coded depth view components, in pixels, in the stream to which the sample entry containing the MVDDepthResolutionBox() is included applies.

`depth_hor_mult_minus1`, `depth_hor_rsh`, `depth_ver_mult_minus1`, and `depth_ver_rsh` are specified identically to the semantics of the syntax elements with the same names in ISO/IEC 14496-10:2014, Annex J. When not present, `depth_hor_mult_minus1` and `depth_ver_mult_minus1` are inferred to be equal to 1, and depth_hor_rsh and `depth_ver_rsh` are inferred to be equal to 0.

NOTE 1   In MVC+D bitstreams, the depth sampling information SEI message carries information corresponding to `depth_hor_mult_minus1`, `depth_hor_rsh`, `depth_ver_mult_minus1`, and `depth_ver_rsh`.

NOTE 2   `depth_hor_mult_minus1` and `depth_hor_rsh` specify the ratio of the width of the luma sample in a texture view component relative to the width of the luma sample in a depth view component. `depth_ver_mult_minus1` and `depth_ver_rsh` specify the ratio of the height of the luma sample in a texture view component relative to the height of the luma sample in a depth view component.

`grid_pos_num_views` specifies the number of views for which `grid_pos_view_id[i]`, `grid_pos_x[grid_pos_view_id[i]]` and `grid_pos_y[grid_pos_view_id[i]]` are present.

`grid_pos_view_id[i]` specifies a view_id value of a texture view.

`grid_pos_x[grid_pos_view_id[i]]` specifies a horizontal offset of a depth sampling grid relative to the luma texture sampling grid in texture luma sample units. grid_pos_y[grid_pos_view_id[i]] specifies a vertical offset of a depth sampling grid relative to the luma texture sampling grid in texture luma sample units. When no value of `grid_pos_view_id[i]` is equal to a `view_id` value of a texture view, `grid_pos_x[view_id]` and `grid_pos_y[view_id]` are inferred to be equal to 0.

NOTE   `grid_pos_num_views, grid_pos_view_id[i], grid_pos_x[grid_pos_view_id[i]]` and `grid_pos_y[grid_pos_view_id[i]]` are specified identically to the semantics of the syntax elements with the same names in ISO/IEC 14496-10:2014, Annex J. In MVC+D bitstreams, the depth sampling information SEI message carries information corresponding to `depth_hor_mult_minus1, depth_hor_rsh, depth_ver_mult_minus1,` and `depth_ver_rsh`.

`mvdDepthRes` contains the width and height of the coded depth view components in the stream to which the sample entry applies. When not present, the width and height of the coded depth view components are inferred to be the same as the width and height of the coded texture view components.

`mvcscalinfosei` contains an SEI NAL unit containing only a view scalability information SEI message as specified in ISO/IEC 14496-10:2014 Annex H. The 'size' field of the container box ViewScalabilityInformationSEIBox shall not be equal to 0 or 1.

`mvdscalinfosei` contains an SEI NAL unit containing only a MVCD scalability information SEI message as specified in ISO/IEC 14496-10:2014 Annex I. The 'size' field of the container box MVDScalabilityInformationSEIBox shall not be equal to 0 or 1.

7.6.4   Sync sample

A sync sample identifies the presence of an IDR access unit of the MVC or MVD bitstream for any sample entry that includes an MVC or MVD configuration record, respectively.

For video data described by a sample entry of type `'mvc3'`, `'mvc4'`, `'mvd3'`, `'mvd4'`, `'a3d3'`, or `'a3d4'`, the following applies:

1.   If the sample is a sync sample, all parameter sets needed for decoding that sample shall be included either in the sample entry or in the sample itself.

2.   Otherwise (the sample is not sync sample), all parameter sets needed for decoding the sample shall be included either in the sample entry or in any of the samples since the previous sync sample to the sample itself, inclusive.

7.6.5   Shadow sync

A shadow sync box shall not be used for video data described by any MVC or MVD sample entry. Its use for MVC or MVD is deprecated.

7.6.6 Independent and disposable samples box

If the `SampleDependencyTypeBox` is used in a track that is both AVC and MVC compatible, then care should be taken that the information provided by this box is true no matter what valid subset of the MVC data (possibly only the AVC data) is used. The 'unknown' values (value 0 of the fields `sample_depends_on`, `sample_is_depended_on`, and `sample_has_redundancy`) may be needed if the information varies.

If the `SampleDependencyTypeBox` is used in a track that is compatible to all of AVC, MVC, and MVD, then care should be taken that the information provided by this box is true no matter what valid subset of the MVD data (possibly only the AVC data or only the MVC data) is used. The 'unknown' values (value 0 of the fields `sample_depends_on`, `sample_is_depended_on`, and `sample_has_redundancy`) may be needed if the information varies.

7.6.7 Sample groups on random access recovery points 'roll' and random access points `'rap '`

When version 0 of the SampleToGroupBox is used, the following applies:

— For video data described by a sample entry of type `'avc1'`, `'avc2'`, `'avc3'` or `'avc4'`, the random access recovery sample group and the random access point sample group identify random access recovery points and random access points, respectively, for all of an AVC decoder, an MVC decoder (if any), and an MVD decoder (if any) operating on the entire bitstream.

   NOTE   If the random access recovery points or the random access points for the AVC decoder, the MVC decoder, and the MVD decoder operating on the entire bitstream are not all aligned, the random access recovery points table or the random access point table, respectively, will not document all of them. In this case, the stream can be stored in multiple tracks, e.g. two tracks, one containing the AVC base layer with a sample entry of type `'avc1'` or `'avc3'`, and the other containing other layers with an MVC or MVD sample entry type.

— For video data described by an MVC sample entry type, the information provided by the random access recovery sample group and the random access point sample group is true for any valid subset of the entire MVC bitstream.

— For video data described by an MVD sample entry type, the information provided by the random access recovery sample group and the random access point sample group is true for any valid subset of in the entire MVD bitstream.

When version 1 of the `SampleToGroupBox` is used, the `grouping_type_parameter` specifies the tier_id value of the layer(s) or view(s) that are refreshed in the associated sample.

7.7 MVC specific information boxes

7.7.1 Overview

The following boxes specify information that relate to more than one output view of an MVC or MVD elementary stream. As any subset of views of an MVC or MVD elementary stream can be chosen for output, the information carried in these boxes is not necessarily specific to any track and thus contained separately. The information can be specified for different groups of output views.

7.7.2   Multiview information box

7.7.2.1   Definition

Box Type: `mvci`

Container: Media Information Box (`'minf'`)

Mandatory: No

Quantity: Zero or one

Located in the Media Information Box of the base view track indicated by the 'sbas' track reference, this box contains Multiview Group boxes, and Multiview Group Relation boxes.

7.7.2.2   Syntax

```
aligned(8) class MultiviewInformationBox

   extends FullBox('mvci', version = 0, flags) {

}
```

7.7.3   Multiview group box

7.7.3.1   Definition

Box Type: `'mvcg'`

Container: Multiview Information box (`'mvci'`)

Mandatory: No

Quantity: Zero or more

This box specifies a multiview group for the views of the MVC or MVD stream that are output. Target output views can be indicated on the basis of track_id, tier_id, or view_id. When the views included in a track match an operating point, it is recommended to use track_id (i.e., entry_type equal to 0) within the Multiview Group box. When multiview sample grouping is in use, and tiers cover more than one view or some tiers contain a temporal subset of the bitstream, it is recommended to use tier_id (i.e., entry_type equal to 1) within the Multiview Group box. Otherwise, it is recommended to use one of the view_id based indications (i.e., entry_type equal to 2 or 3).

When entry_type is equal to 0 or 1, the following applies. Each view in a track or tier that is included in this box is a target output view, and if a track or tier included in this box contains multiple views, all the contained views are target output views.

Decoding of the output views may require decoding of other views that are not target output views. The views that are required for decoding but are not target output views can be concluded from reference view identifiers included in the View Identifier box, the `'scal'` track references, or from the Tier Dependency box.

If the box contains a track_id or tier_id that is not present or refers to a view_id of a view that is not present, the respective view should be considered removed and the multiview group should be ignored.

7.7.3.2 Syntax

```
aligned(8) class MultiviewGroupBox extends FullBox('mvcg', version = 0, flags) {
    unsigned int(32) multiview_group_id;
    unsigned int(16) num_entries;
    bit(8) reserved = 0;
    for(i=0; i<num_entries; i++) {
        unsigned int(8) entry_type;
        if (entry_type == 0)
            unsigned int(32) track_id;
        else if (entry_type == 1) {
            unsigned int(32) track_id;
            unsigned int(16) tier_id;
        }
        else if (entry_type == 2) {
            bit(6) reserved1 = 0;
            unsigned int(10) output_view_id;
        }
        else if (entry_type == 3) {
            bit(6) reserved2 = 0;
            unsigned int(10) start_view_id;
            unsigned int(16) view_count;
        }
    }
    TierInfoBox subset_stream_info;       // optional
    MultiviewRelationAttributeBox relation_attributes; // optional
    TierBitRateBox subset_stream_bit_rate; // optional
    BufferingBox subset_stream_buffering;  // optional
    MultiviewSceneInfoBox multiview_scene_info;       // optional
}
```

7.7.3.3 Semantics

`multiview_group_id` provides a unique identifier for the multiview group within the file.

`num_entries` is the number of tracks (entry type 0), tiers (entry type 1), target output views (entry type 2), or continuous sets of target output views (entry type 3) included in this multiview group.

`entry_type` specifies how the target output views are indicated. The following values of entry_type are specified:

— 0 – all the views included in an indicated track are target output views

— 1 – the view(s) of an indicated tier within an indicated track are target output views

— 2 – the view with view_id equal to output_view_id is a target output view

— 3 – the views having view_id within the range of start_view_id to (start_view_id + view_count – 1), inclusive, are target output views

`track_id` indicates a track containing target output views.

`tier_id` indicates a tier within a track where all views within the tier are target output views.

`output_view_id` indicates a view_id of a target output view.

`start_view_id` indicates the first view_id in a range of contiguous values of view_id all being target output views.

`view_count` indicates the number of contiguous values of view_id all being target output views.

`track_id` indicates a track.

`tier_id` indicates a tier within a track.

`subset_stream_info` indicates the characteristics of the bitstream subset containing the indicated output views and the views they depend on.

`relation_attributes` indicate the relations between output views. If `'ecam'` is used as a common attribute, all the output views are associated with extrinsic camera parameters indicating that the cameras have identical rotation and constant spacing. If `'ecam'` is used as a differentiating attribute, at least one output view is associated with extrinsic camera parameters with different rotation from the others or the output views are associated with extrinsic camera parameters not having a constant spacing.

`subset_stream_bit_rate` indicates the bit rate statistics of the bitstream subset containing the indicated output views and the views they depend on. The values of `tierBaseBitRate`, `tierMaxBitRate`, and `tierAvgBitRate` within the TierBitRateBox are unspecified.

`subset_stream_buffering` indicates the HRD parameters that apply to the bitstream subset containing the indicated output views and the views they depend on and operating with the indicated target output views.

`multiview_scene_info` contains the maximum disparity in units of integer pixel resolution between any spatially adjacent output views in any access unit.

7.7.4 Multiview group relation box

7.7.4.1 Definition

Box Type: `'swtc'`

Container: Multiview Information box (`'mvci'`)

Mandatory: No

Quantity: Zero or more

This box specifies a set of multiview groups from which one multiview group is decoded and played at any time. The given relation attributes specify which features are common in all associated multiview groups and which factors make the multiview groups differ from each other. The relation attributes can be used to select a suitable set of multiview groups for playback, e.g., based on the number of output views. The differentiating attributes can be used to select which multiview group within the set is suitable for the player, e.g., based on the required level for decoding.

7.7.4.2 Syntax

```
aligned(8) class MultiviewGroupRelationBox() extends FullBox('swtc', version = 0, flags) {
    unsigned int(32) num_entries;
    for (i=0; i<num_entries; i++)
        unsigned int(32) multiview_group_id;
    MultiviewRelationAttributeBox relation_attributes;
}
```

7.7.4.3 Semantics

`num_entries` indicates the number of associated multiview groups.

`multiview_group_id` is the identifier of an associated multiview group.

`relation_attributes` indicate the relations between the associated multiview groups.

7.7.5 Multiview relation attribute box

7.7.5.1 Definition

Box Type: `mvra'

Container: MultiviewGroupBox or MultiviewGroupRelationBox

Mandatory: No in MultiviewGroupBox, Yes in MultiviewGroupRelationBox

Quantity: Zero or One in MultiviewGroupBox

              One in MultiviewGroupRelationBox

When the Multiview Relation Attribute box is contained in a Multiview Group box, it indicates the relation of the output views of the respective multiview group with each other. When the Multiview Relation Attribute box is contained in a Multiview Group Relation box, it indicates the relation of the multiview groups with each other.

The Multiview Relation Attribute box contains common and differentiating attributes. When the Multiview Relation Attribute box is included in a Multiview Group box, a common attribute indicates a characteristic that is common for each one of the target output views of the multiview group and a differentiating attribute indicates a characteristic that differs in at least one of one of the target output views of the multiview group. When Multiview Relation Attribute box is included in a Multiview Group Relation box, a common attribute indicates a characteristic that is common for the indicated multiview groups or for the respective target output views in each one of the indicated multiview groups, whereas a differentiating attribute indicates a characteristic that differs in at least one of the indicated multiview groups or at least one of the respective target output views in the indicated multiview groups.

A common attribute is associated with an additional parameter, which carries the value of the common attribute. The syntax and semantics of the additional parameter depend on the attribute in question.

For example, a file writer can create a Multiview Group for each stereo pair suitable for display from a multiview bitstream. Furthermore, a file writer can create a Multiview Group Relation box listing all the multiview groups for stereo pair output and including a Multiview Relation Attribute box with common attributes number of views (equal to 2) and in-line camera arrangement. A file reader can study the Multiview Group Relation box to find the options for stereo pair output and choose one multiview group for processing. Note that the presence of views in a group does not necessarily imply they are all suggested as output views at any given time – the terminal may choose which views to output, and it is not limited by the group information.

7.7.5.2 Syntax

```
aligned(8) class MultiviewRelationAttributeBox

    extends FullBox('mvra', version = 0, flags) {

    bit(16) reserved1 = 0;

    unsigned int(16) num_common_attributes;

    for (i=0; i<num_common_attributes; i++) {

        unsigned int(32) common_attribute;

        unsigned int(32) common_value;

    }

    bit(16) reserved2 = 0;

    unsigned int(16) num_differentiating_attributes;

    for (i=0; i<num_differentiating_attributes; i++)

        unsigned int(32) differentiating_attribute;

}
```

7.7.5.3 Semantics

common_attribute and differentiating_attribute are selected from the list below. Attributes that can be used as a differentiating attribute are associated with a distinguishing pointer to the field or information.

common_value specifies the value for the common attribute. Its syntax and semantics depend on the common attribute and are specified in the table below.

Name	Attribute	Pointer and semantics	common_value syntax and semantics

Profile	'prfl'	This attribute shall not be included in the Multiview Group box.
		
		When included in the Multiview Group Relation box, the attribute refers to the profile required for decoding the bitstream subset corresponding to the multiview group. The attribute points to the profileIndication field of the subset_stream_info element of the Multiview Group box.
			`bit(24) reserved = 0;`
			`unsigned int(8)`
			`profileIndication;`
			
			profileIndication is the profile sufficient for decoding the bitstream subset corresponding to all indicated multiview groups.

Level	'levl'	This attribute shall not be included in the Multiview Group box.
		
		When included in the Multiview Group Relation box, the attribute refers to the level required for decoding the bitstream subset corresponding to the multiview group. The attribute points to the levelIndication field of the subset_stream_info element of the Multiview Group box.
			`bit(24) reserved = 0;`
			`unsigned int(8)`
			`levelIndication;`
			
			profileIndication is the level sufficient for decoding the bitstream subset corresponding to all indicated multiview groups, or 0 if the level is unspecified.

Bitrate	'bitr'	This attribute shall not be included in the Multiview Group box. When included in the Multiview Group Relation box, the attribute refers to the total size of bitstream subset required for decoding of the multiview group divided by the duration in the track header box. The attribute points to the avgBitRate field of the subset_stream_bit_rate element of the Multiview Group box, if present, or a value that would be contained in the avgBitRate field of the subset_stream_bit_rate element of the Multiview Group box, if it were present.	`unsigned int(32) bitrate;` bitrate indicates the average bit rate in bits per second of the bitstream subset required for decoding the multiview group. The bitrate may be rounded up.

Frame rate	'frar'	This attribute shall not be included in the Multiview Group box.
		
		When included in the Multiview Group Relation box, the attribute refers the number of samples in the track divided by duration in the track header box.
			`unsigned int(16) integer_part;`
			`bit(16) reserved = 0;`
			
			integer_part shall be equal to the output rate of decoded access units in second rounded to the closest integer using the Round function specified in ISO/IEC 14496-10.

Number of output views	'nvws'	Number of target output views indicated in the Multiview Group Box ('mvcg') If this attribute is included in the Multiview Group box, it shall be a common attribute and merely documents the number of output views in the respective multiview group.	`unsigned int(32) num_views;` num_views indicates the number of views in the multiview group.
Intrinsic camera parameters	'icam'	The intrinsic camera parameters are stored in 'avc1', 'avc2', 'avc3', 'avc4', 'mvc1', 'mvc2', 'mvc3', 'mvc4', 'mvd1', 'mvd2', 'mvd3', 'mvd4', 'a3d1', 'a3d2', 'a3d3', or 'a3d4' Sample Entry (in Sample entry box of media track). If this attribute is included in the Multiview Group box and used as a common attribute, the intrinsic camera parameters of the target output views are identical. If this attribute is included in the Multiview Group box and used as a differentiating attribute, the intrinsic camera parameters of the target output views differ at least partly. If this attribute is included in the Multiview Group Relation box and used as a common attribute, the number of target output views in all indicated multiview groups shall be the same and the intrinsic camera parameters of the respective target output views in all indicated multiview groups are identical. If this attribute is included in the Multiview Group Relation box and used as a differentiating attribute, the intrinsic camera parameters of the respective target output views differ at least partly.	Unspecified.
Extrinsic camera parameters	'ecam'	The extrinsic camera parameters are stored in 'avc1', 'avc2', 'avc3', 'avc4', 'mvc1', 'mvc2', 'mvc3', 'mvc4', 'mvd1', 'mvd2', 'mvd3', 'mvd4', 'a3d1', 'a3d2', 'a3d3', or 'a3d4' Sample Entry (in Sample entry box of media track).	Unspecified.

If this attribute is included in the Multiview Group box and used as a common attribute, the rotation of the cameras for all the target output views is the same and, if the cameras are arranged in linear, elliptical, or rectangular arrangement, the distance of adjacent cameras is the same. If this attribute is included in the Multiview Group box and used as a differentiating attribute, the rotation or the distance of adjacent cameras in linear, elliptical, or rectangular arrangement differs.

If the attribute is included in the Multiview Group Relation box and used as a common attribute, the relative extrinsic camera parameters target output views in all indicated multiview groups are identical. That is, the distance of cameras relative to each other and their rotation matches in the indicated multiview groups. If the attribute is included in the Multiview Group Relation box and used as a differentiating attribute, the relative extrinsic camera parameters of respective target output views differ at least partly.

Inline view array	'ilvi'	If used as a common attribute, the associated cameras are located on a straight line.	

When included in a Multiview Group box, the attribute shall be a common attribute.

```
bit(28) reserved = 0;
unsigned int(2) horizontal_
order;
unsigned int(2) vertical_order;
```

horizontal_order
    indicates the horizontal order of the views:

0:    the views are in the same horizontal location

1:    the views are ordered left-to-right

2:    the views are ordered right-to-left

3:    the order of the views is undefined, or left and right are not well-defined.

			`vertical_order` indicates the vertical order of the views:
			0: the views are in the same vertical location
			1: the views are ordered bottom-to-top
			2: the views are ordered top-to-bottom
			3: the order of the views is undefined, or top and bottom are not well-defined.

Rectangular view array	`'rtvi'`	If used as a common attribute, the associated cameras form a rectangular shape and are regularly spaced along the orthogonal coordinate axes.
		
		When included in a Multiview Group box, the attribute shall be a common attribute.
			`unsigned int(16) row_view_count;`
			`unsigned int(16) col_view_count;`
			
			`row_view_count`
			    specifies the number of rows in the rectangular array.
			
			`col_view_count` specifies the number of columns in the rectangular array.
			
			The views are indicated in raster scan order in the Multiview Group box.

Planar view array	`'plvi'`	If used as a common attribute, the associated cameras are located on a plane, but may be irregularly spaced. When included in a Multiview Group box, the attribute shall be a common attribute.	Unspecified.

Elliptical view array	`'elvi'`	If used as a common attribute, the associated cameras are located on the arc of an ellipse.
		
		When included in a Multiview Group box, the attribute shall be a common attribute.
			`bit(28) reserved = 0;`
			`unsigned int(2) horizontal_order;`
			`unsigned int(2) vertical_order;`
			
			The semantics are identical to those for the Inline view array.

Spherical view array	'spvi'	If used as a common attribute, the associated cameras are located on the surface of a sphere. When included in a Multiview Group box, the attribute shall be a common attribute.	Unspecified.

Stereo view array	'stvi'	If used as a common attribute, the associated cameras are a pair of views suitable for stereo viewing.
		
		When included in a Multiview Group box, the attribute shall be a common attribute.
			`bit(6) reserved1 = 0;`
			`unsigned int(10) left_view_id;`
			`bit(6) reserved2 = 0;`
			`unsigned int(10) right_view_id;`

Geometry	'geom'	If used as a differentiating attribute, indicates that the views or groups of views belong to different view arrangements (e.g. inline, planar, etc.)	Unspecified.

7.7.6 Multiview scene info box

7.7.6.1 Definition

Box Type: 'vwdi'

Container: Multiview Group box ('mvcg')

Mandatory: No

Quantity: Zero or one

An optional Multiview Scene Info Box includes the maximum disparity between the adjacent views of the respective multiview group. This information can be used for processing the multiview video prior to rendering on a 3D display.

NOTE   A Multiview Scene Information SEI message, as specified in MVC H.12.1.5, can indicate the maximum disparity between any adjacent views in the bitstream. Thus, the Multiview Scene Info Box represents similar information as carried in the Multiview Scene Information SEI message but is limited to a certain set of views rather than concerns all the views in the bitstream.

The Multiview Scene Info Box shall not be present for multiview groups associated with cameras that do not form a one-dimensional arrangement, such as a line or an arc of an ellipse.

7.7.6.2 Syntax

```
class MultiviewSceneInfoBox extends Box ('vwdi')

{

   unsigned int(8)   max_disparity;

}
```

7.7.6.3 Semantics

max_disparity specifies the maximum disparity in units of integer luma samples between the spatially adjacent view components (within an access unit) in this multiview group. This information can be used for processing the multiview video prior to rendering on a 3D display.

7.7.7 MVC view priority assignment box

7.7.7.1 Definition

Box Type: 'mvcP'

Container: Sample Entry ('avc1', 'avc2', 'avc3', 'avc4',
                            'mvc1', 'mvc2', 'mvc3', 'mvc4',
                            'mvd1', 'mvd2', 'mvd3', 'mvd4')

Mandatory: No

Quantity: Zero or one

A priority assignment URI provides the name (in the URI space) of a method used to assign content priority values in the View Priority sample grouping. The URI is treated here as a name only; it should be de-referenceable, though this is not required. File readers may be able to recognize some methods and thereby know what stream extraction or selection of output views based on particular content priority values would do.

7.7.7.2 Syntax

```
class MVCViewPriorityAssignmentBox extends Box('mvcP')
{
    unsigned int(8)   method_count;
    string PriorityAssignmentURI[method_count];
}
```

7.7.7.3 Semantics

method_count provides a count of the number of following URIs.

PriorityAssignmentURI provides a unique name of the method used to assign content_priority_id values in View Priority sample groupings. In the case of absence of this box, the priority assignment method is unknown.

8 HEVC elementary streams and sample definitions

8.1 Overview

The High Efficiency Video Coding (HEVC) standard, jointly developed by the ITU-T and ISO/IEC JTC 1/SC 29/WG 11 (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 clause specifies the storage format for single-layer HEVC (ISO/IEC 23008-2) video streams.

The storage of HEVC content uses the existing capabilities of the ISO base media file format but also defines extensions to support the following features of the HEVC codec.

a)   Parameter sets:

The video, sequence and picture parameter set mechanism decouples the transmission of infrequently changing information from the transmission of coded block data. Each slice containing the coded block 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, and the sequence parameter set references a video parameter set that contains global decoding parameter information (across layers or view in potential scalable and 3DV extensions).

This document includes the following tools for supporting of HEVC contents:

a)   Temporal scalability sample grouping:

a structuring and grouping mechanism to indicate the association of access units with different hierarchy levels of temporal scalability.

b)   Temporal sub-layer access sample grouping:

a structuring and grouping mechanism to indicate the identification of access units as temporal sub-layer access (TSA) samples.

c)   Step-wise temporal sub-layer access sample grouping:

a structuring and grouping mechanism to indicate the identification of access units as step-wise temporal sub-layer access (STSA) samples.

8.2   Elementary stream structure

In this clause, a video stream is represented by one video track in a file.

Two types of elementary streams are defined for storing HEVC content:

—   **Video Elementary Stream** that does not contain any parameter sets; all parameter sets are stored in a sample entry or sample entries;

—   **Video and Parameter set elementary stream** that may contain parameter sets, and may also have parameter sets stored in their sample entry or sample entries.

8.3   Sample and configuration definition

8.3.1   Overview

HEVC sample: An HEVC sample is an access unit as defined in ISO/IEC 23008-2:2017, 3.1.

NOTE 2   Some HEVC streams as defined in ISO/IEC 23008-2 include data that can be stored alternatively according to Clause 9.

8.3.2   Canonical order and restrictions

The canonical stream format is an HEVC elementary stream that satisfies the following conditions in addition to the general conditions in subclause 4.3.2:

—   **Access unit delimiter NAL units**: The constraints obeyed by access unit delimiter NAL units are defined in ISO/IEC 23008-2.

—   **Parameter sets**: A parameter set to be used in a picture must be sent prior to the sample containing that picture or in the sample for that picture. For a video stream that a particular sample entry

applies to, the video parameter set, sequence parameter sets, and picture parameter sets, shall be stored only in the sample entry when the sample entry name is `'hvc1'`, and may be stored in the sample entry and the samples when the sample entry name is `'hev1'`.

NOTE 1   Storing parameter sets in the sample entries of a video stream provides a simple and static way to supply parameter sets. Storing parameters in samples on the other hand is more complex but allows for more dynamism in the case of parameter set updates (a particular parameter set's content is changed but using the same ID) and in the case of adding additional parameter sets. A decoder initializes with the parameter sets in the sample entry, and then updates using the parameter sets as they occur in the stream, starting from any sample marked as a sync sample. Such updating can replace parameter sets with a new definition using the same identifier. Each time the sample entry changes, the decoder re-initializes with the parameter sets included in the sample entry.

— **SEI messages:** SEI messages of declarative nature may be stored in the sample entry; there is no prescription about removing such SEI messages from the samples.

— **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 when the sample entry does not also permit in-stream parameter sets.

NOTE 2   The removal or addition of Filler Data NAL units, start codes, SEI messages or Filler Data SEI messages can change the bitstream characteristics with respect to conformance with the HRD when operating the HRD in CBR mode as specified in ISO/IEC 23008-2:2017, Annex C.

8.3.3   Decoder configuration information

8.3.3.1   HEVC decoder configuration record

8.3.3.1.1   Definition

This subclause specifies the decoder configuration information for ISO/IEC 23008-2 video content.

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 parameter sets, if stored in the sample entry. This record is externally framed (its size is supplied by the structure that contains it).

This record contains a version field. This document defines version 1 of this record. Incompatible changes to the record will be indicated by a change of version number. Readers shall 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.

The values for `general_profile_space`, `general_tier_flag`, `general_profile_idc`, `general_profile_compatibility_flags`, `general_constraint_indicator_flags`, `general_level_idc`, `min_spatial_segmentation_idc`, `chroma_format_idc`, `bit_depth_luma_minus8` and `bit_depth_chroma_minus8` shall be valid for all parameter sets that are activated when the stream described by this record is decoded (referred to as "all the parameter sets" in the following sentences in this paragraph). Specifically, the following restrictions apply:

— The value of `general_profile_space` in all the parameter sets shall be identical.

— The tier indication `general_tier_flag` shall indicate a tier equal to or greater than the highest tier indicated in all the parameter sets.

— The profile indication `general_profile_idc` shall indicate a profile to which the stream associated with this configuration record conforms.

NOTE 1   If the sequence parameter sets are marked with different profiles, then the stream could need examination to determine which profile, if any, the entire stream conforms to. If the entire stream is not examined, or the examination reveals that there is no profile to which the entire stream conforms, then the entire stream shall be split into two or more sub-streams with separate configuration records in which these rules can be met.

— Each bit in `general_profile_compatibility_flags` may only be set if all the parameter sets set that bit.

— Each bit in `general_constraint_indicator_flags` may only be set if all the parameter sets set that bit.

— The level indication `general_level_idc` shall indicate a level of capability equal to or greater than the highest level indicated for the highest tier in all the parameter sets.

— The `min_spatial_segmentation_idc` indication shall indicate a level of spatial segmentation equal to or less than the lowest level of spatial segmentation indicated in all the parameter sets.

— The value of `chroma_format_idc` in all the parameter sets shall be identical.

— The value of `bit_depth_luma_minus8` in all the parameter sets shall be identical.

— The value of `bit_depth_chroma_minus8` in all the parameter sets shall be identical.

Explicit indication is provided in the HEVC Decoder Configuration Record about the chroma format and bit depth as well as other important format information used by the HEVC video elementary stream. Each type of such information shall be identical in all parameter sets, if present, in a single HEVC configuration record. If two sequences differ in any type of such information, two different HEVC sample entries shall be used. If the two sequences differ in color space indications in their VUI information, then two different HEVC sample entries are also required.

There is a set of arrays to carry initialization NAL units. The NAL unit types are restricted to indicate SPS, PPS, VPS, prefix SEI, and suffix SEI NAL units only. NAL unit types that are reserved in ISO/IEC 23008-2 and in this document may acquire a definition in future, and readers should ignore arrays with reserved or unpermitted values of NAL unit type.

NOTE 2   This 'tolerant' behaviour is designed so that errors are not raised, allowing the possibility of backwards-compatible extensions to these arrays in future specifications.

It is recommended that the arrays be in the order VPS, SPS, PPS, prefix SEI, suffix SEI.

When `general_non_packed_constraint_flag` (bit 3 of the 6-byte general_constraint_indicator_flags) is equal to 0 and some of the samples referring to this sample entry represent frame-packed content and any of the default display windows specified by the active SPSs for the samples referring to this sample entry covers more than one constituent frame of the frame-packed content, the techniques described in ISO/IEC 14496-12:2015, 8.15 using the scheme type "stvi" shall be used. In this case, the `stereo_scheme` in the Stereo Video Box should be set to 1, to indicate that the frame packing scheme used in HEVC is the same as in AVC.

8.3.3.1.2   Syntax

```
aligned(8) class HEVCDecoderConfigurationRecord {

   unsigned int(8) configurationVersion = 1;

   unsigned int(2) general_profile_space;

   unsigned int(1) general_tier_flag;

   unsigned int(5) general_profile_idc;

   unsigned int(32) general_profile_compatibility_flags;

   unsigned int(48) general_constraint_indicator_flags;
```

```
unsigned int(8) general_level_idc;

bit(4) reserved = '1111'b;

unsigned int(12) min_spatial_segmentation_idc;

bit(6) reserved = '111111'b;

unsigned int(2) parallelismType;

bit(6) reserved = '111111'b;

unsigned int(2) chroma_format_idc;

bit(5) reserved = '11111'b;

unsigned int(3) bit_depth_luma_minus8;

bit(5) reserved = '11111'b;

unsigned int(3) bit_depth_chroma_minus8;

unsigned int(16) avgFrameRate;

unsigned int(2) constantFrameRate;

unsigned int(3) numTemporalLayers;

unsigned int(1) temporalIdNested;

unsigned int(2) lengthSizeMinusOne;

unsigned int(8) numOfArrays;

for (j=0; j < numOfArrays; j++) {

    unsigned int(1) array_completeness;

    bit(1) reserved = 0;

    unsigned int(6) NAL_unit_type;

    unsigned int(16) numNalus;

    for (i=0; i< numNalus; i++) {

        unsigned int(16) nalUnitLength;

        bit(8*nalUnitLength) nalUnit;

    }

}

}
```

8.3.3.1.3 Semantics

general_profile_space,   general_tier_flag,   general_profile_idc,   general_profile_
compatibility_flags, general_constraint_indicator_flags, general_level_idc, min_spatial_
segmentation_idc,   chroma_format_idc,   bit_depth_luma_minus8 and bit_depth_chroma_minus8
contain the matching values for the fields general_profile_space, general_tier_flag, general_profile_
idc, general_profile_compatibility_flag[i] for i from 0 to 31, inclusive, the 6 bytes starting with the
byte containing the general_progressive_source_flag, general_level_idc, min_spatial_segmentation_

idc, chroma_format_idc, bit_depth_luma_minus8 and bit_depth_chroma_minus8 as defined in ISO/IEC 23008-2, for the stream to which this configuration record applies.

parallelismType indicates the type of parallelism that is used to meet the restrictions imposed by min_spatial_segmentation_idc when the value of min_spatial_segmentation_idc is greater than 0. Value 1 indicates that the stream to which this configuration record applies supports slice based parallel decoding. Value 2 indicates that the stream to which this configuration record applies supports tile based parallel decoding. Value 3 indicates that the stream to which this configuration record applies supports entropy coding synchronization based parallel decoding. Value 0 indicates that the stream supports mixed types of parallel decoding or that the parallelism type is unknown. The values above can be inferred by the fields tiles_enabled_flag and entropy_coding_sync_enabled_flag as defined in ISO/IEC 23008-2. Specifically: if the fields tiles_enabled_flag and entropy_coding_sync_enabled_flag are both equal to 0 in all PPSs that are activated when the stream to which this configuration record applies is decoded, then parallelismType can be set to 1. If the field tiles_enabled_flag is equal to 1 in all PPSs that are activated when the stream to which this configuration record applies is decoded, then parallelismType can be set to 2. If the field entropy_coding_sync_enabled_flag is equal to 1 in all PPSs that are activated when the stream to which this configuration record applies is decoded, then parallelismType can be set to 3. If none of the above is true (or if it is unknown which of them is true) then parallelismType should be set to 0.

avgFrameRate gives the average frame rate in units of frames/(256 seconds), for the stream to which this configuration record applies. Value 0 indicates an unspecified average frame rate.

constantFrameRate equal to 1 indicates that the stream to which this configuration record applies is of constant frame rate. Value 2 indicates that the representation of each temporal layer in the stream is of constant frame rate. Value 0 indicates that the stream may or may not be of constant frame rate.

numTemporalLayers greater than 1 indicates that the stream to which this configuration record applies is temporally scalable and the contained number of temporal layers (also referred to as temporal sub-layer or sub-layer in ISO/IEC 23008-2) is equal to numTemporalLayers. Value 1 indicates that the stream is not temporally scalable. Value 0 indicates that it is unknown whether the stream is temporally scalable.

temporalIdNested equal to 1 indicates that all SPSs that are activated when the stream to which this configuration record applies is decoded have sps_temporal_id_nesting_flag as defined in ISO/IEC 23008-2 equal to 1 and temporal sub-layer up-switching to any higher temporal layer can be performed at any sample. Value 0 indicates that the conditions above are not or may not be met.

lengthSizeMinusOne plus 1 indicates the length in bytes of the NALUnitLength field in an HEVC video sample in the stream to which this configuration record applies. 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.

numArrays indicates the number of arrays of NAL units of the indicated type(s).

array_completeness when equal to 1 indicates that all NAL units of the given type are in the following array and none are in the stream; when equal to 0 indicates that additional NAL units of the indicated type may be in the stream; the default and permitted values are constrained by the sample entry name.

NAL_unit_type indicates the type of the NAL units in the following array (which shall be all of that type); it takes a value as defined in ISO/IEC 23008-2; it is restricted to take one of the values indicating a VPS, SPS, PPS, prefix SEI, or suffix SEI NAL unit.

numNalus indicates the number of NAL units of the indicated type included in the configuration record for the stream to which this configuration record applies. The SEI array shall only contain SEI messages of a 'declarative' nature, that is, those that provide information about the stream as a whole. An example of such an SEI could be a user-data SEI.

nalUnitLength indicates the length in bytes of the NAL unit.

nalUnit contains an SPS, PPS, VPS or declarative SEI NAL unit, as specified in ISO/IEC 23008-2.

8.4   Derivation from ISO base media file format

8.4.1   HEVC video stream definition

8.4.1.1   Sample entry name and format

8.4.1.1.1   Definition

Sample Entry and Box Types: 'hvc1', 'hev1', 'hvcC'

Container: Sample Table Box ('stbl')

Mandatory: An 'hvc1' or 'hev1' sample entry is mandatory

Quantity: One or more sample entries may be present

An HEVC visual sample entry shall contain an HEVC Configuration Box, as defined below. This includes an HEVCDecoderConfigurationRecord, as defined in subclause 8.3.3.1.

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

Multiple sample entries may be used, as permitted by ISO/IEC 14496-12, to indicate sections of video that use different configurations or parameter sets.

When the sample entry name is 'hvc1' or 'hev1', the stream to which this sample entry applies shall be a compliant and HEVC stream as viewed by an HEVC decoder operating under the configuration (including profile, tier, and level) given in the HEVCConfigurationBox.

When the sample entry name is 'hvc1', the default and mandatory value of array_completeness is 1 for arrays of all types of parameter sets, and 0 for all other arrays. When the sample entry name is 'hev1', the default value of array_completeness is 0 for all arrays.

8.4.1.1.2   Syntax

```
class HEVCConfigurationBox extends Box('hvcC') {

    HEVCDecoderConfigurationRecord() HEVCConfig;

}

class HEVCSampleEntry() extends VisualSampleEntry ('hvc1' or 'hev1'){

    HEVCConfigurationBox config;

    MPEG4ExtensionDescriptorsBox (); // optional

}
```

8.4.1.1.3   Semantics

Compressorname in the base class VisualSampleEntry indicates the name of the compressor used with the value "\013HEVC Coding" being recommended (\013 is 11, the length of the string in bytes).

HEVCDecoderConfigurationRecord is defined in subclause 8.3.3.

8.4.2 Parameter sets in sample entry

This subclause applies to a particular type of parameter sets (VPSs, SPSs, or PPSs) when the particular type of parameter sets is included in the sample entry.

Each HEVC sample entry, which contains the HEVC video stream decoder specific information, includes a group of the particular type of parameter sets. 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. When the value of the applicable `array_completeness` is 1, a parameter set cannot be updated without causing a different sample entry to be used.

Systems wishing to send parameter set 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, the parameter sets are included in the samples of the stream.

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

8.4.3 Sync sample

A sync sample in `'hvc1'` and `'hev1'` tracks shall contain VCL NAL units indicating that the coded picture with nuh_layer_id equal to 0 in the sample is an Instantaneous Decoding Refresh (IDR) picture, a Clean Random Access (CRA) picture, or a Broken Link Access (BLA) picture. When the coded picture with nuh_layer_id equal to 0 in a sync sample is a BLA or CRA picture, there shall be no RASL pictures associated with that BLA or CRA picture.

Former versions of this document allowed the presence of RASL pictures associated with sync samples, although this is not in conformance with ISO/IEC 14496-12. When RASL pictures are associated with a sync sample, tools that inspect only file-format structures might reach incorrect conclusions about the dependencies in the bitstream. RASL pictures can be signalled in the file format as leading pictures in the SampleDependencyTypeBox, and BLA and CRA pictures with RASL pictures can be signalled as a SAP of type 3, using the SAP sample group or using the RAP sample group. For older files conforming to previous version of the specification, when performing random access into this track, or after a discontinuous edit, the flag *HandleCraAsBlaFlag* can be set for the HEVC decoder. For environments where HEVC file processors need information that sync samples are not associated with RASL pictures, the brand `'nras'`, as defined in Annex D.5, may be used to indicate compliance to this and later versions of this document.

NOTE   In the context of L-HEVC file format specified in Clause 9, the signalling of sync samples that concerns a particular base layer picture may be different from the signalling of sync samples here for the same picture.

Table 10 indicates the mapping between HEVC VCL NAL unit types, ISOBMFF sync sample status and SAP types as documented in ISOBMFF.

Table 10 — Mapping of sync sample status and SAP types to NAL unit type

NAL Unit Type	ISOBMFF sync sample status	DASH SAP type
IDR_N_LP	true	1
IDR_W_RADL	true	2 (if the IRAP has associated RADL pictures) 1 (if the IRAP has no associated RADL pictures)
BLA_N_LP	true	1
(*1) see above discussion on sync samples and leading HEVC pictures		

Table 10 (continued)

NAL Unit Type	ISOBMFF sync sample status	DASH SAP type
BLA_W_RADL	true	2 (if the IRAP has associated RADL pictures) 1 (if the IRAP has no associated RADL pictures)
BLA_W_LP	false (*1)	3 (if the IRAP has associated RASL pictures)
	true	2 (if the IRAP has no associated RASL pictures but has associated RADL pictures)
	true	1 (if the IRAP has no associated leading pictures)
CRA	false (*1)	3 (if the IRAP has associated RASL pictures)
	true	2 (if the IRAP has no associated RASL pictures but has associated RADL pictures)
	true	1 (if the IRAP has no associated leading pictures)
(*1) see above discussion on sync samples and leading HEVC pictures		

When the sample entry name is 'hev1', the following applies:

— If the sample is a sync sample, all parameter sets needed for decoding that sample shall be included either in the sample entry or in the sample itself.

— Otherwise (the sample is not a sync sample), all parameter sets needed for decoding the sample shall be included either in the sample entry or in any of the samples since the previous sync sample to the sample itself, inclusive.

For signalling of various types of random access points, the following guidelines are recommended:

— The sync sample table (and the equivalent flag in movie fragments) shall be used in an HEVC track unless all samples are sync samples. Note that track fragment random access box refers to the presence of signalled sync samples in a movie fragment.

— The 'roll' sample group is recommended to be used only for gradual decoding refresh (GDR) based random access points, i.e. those that contain non-intra coded slices.

— The use of the 'rap ' or 'sync' sample group is optional, depending on the need of either the information on leading samples associated with the random access points or the picture types (e.g. IDR, CRA, or BLA) of the random access points.

— The use of the Alternative Startup Sequences (ISO/IEC 14496-12:2015, 10.3) sample group is recommended to be used only with random access points consisting of CRA and BLA pictures.

In the context of this clause, the leading samples, defined as part of the definition of the 'rap ' sample group in ISO/IEC 14496-12, contain Random Access Skipped Leading (RASL) access units as defined in ISO/IEC 23008-2.

8.4.4   Sync sample sample grouping

8.4.4.1   Overview

Sync samples in HEVC may be of various types. These sample groups may be used to identify the sync samples of a specific type. If a sample group is given for a specific type of sync sample, then all samples (if any) containing that type of sync sample are marked by the group. If the group is absent (there is no sample to group mapping for that type), it is unknown which samples contain a sync sample of that type.

8.4.4.2 Sync sample sample group entry

8.4.4.2.1 Definition

Group Types: `'sync'`

Container: Sample Group Description Box (`'sgpd'`)

Mandatory: No

Quantity: Zero or more

A sync sample sample group entry identifies samples containing a sync sample of a specific type. The `grouping_type_parameter` is not defined for the `SampleToGroupBox` with grouping type `'sync'`.

8.4.4.2.2 Syntax

```
class SyncSampleEntry() extends VisualSampleGroupEntry ('sync')

{

    bit(2) reserved = 0;

    unsigned int(6) NAL_unit_type;

}
```

8.4.4.2.3 Semantics

`NAL_unit_type` shall be a type that identifies a valid sync sample (e.g. IDR).

8.4.5 Temporal scalability sample grouping

8.4.5.1 Overview

An HEVC video track may contain zero or one instance of a `SampleToGroupBox` with a `grouping_type` `'tscl'`. This `SampleToGroupBox` instance represents the assignment of samples in the track to temporal layers (referred to as temporal sub-layers or sub-layers in ISO/IEC 23008-2). An accompanying instance of the SampleGroupDescriptionBox with the same grouping type shall be present, and contain TemporalLayerEntry sample group entries describing the temporal layers.

There may also be a set of Temporal Level sample group definitions (ISO/IEC 14496-12:2015, 10.5). A sample mapped to a sample group description entry with index A of a Temporal Level sample grouping shall also be considered mapped to the sample group description entry in the temporal scalability sample group (defined here) having `temporalLayerId` equal to A (i.e. a sample to group mapping for the temporal scalability sample group may also be present but is not required).

8.4.5.2 Temporal layer sample group entry

8.4.5.2.1 Definition

Group Types: `'tscl'`

Container: Sample Group Description Box (`'sgpd'`)

Mandatory: No

Quantity: Zero or more

A temporal layer sample group entry defines the temporal layer information for all samples in a temporal layer. Temporal layers are numbered with non-negative integers, and each temporal layer is associated with a particular value of `TemporalId` as defined in ISO/IEC 23008-2. A temporal layer associated with a `TemporalId` value greater than 0 depends on all temporal layers associated with lower `TemporalId` values. A temporal layer representation, also referred to as the representation of a temporal layer, associated with a particular `TemporalId` value consists of all temporal layers associated with `TemporalId` values less than or equal to the given TemporalId value.

The `grouping_type_parameter` is not defined for the `SampleToGroupBox` with grouping type `'tscl'`.

8.4.5.2.2 Syntax

```
class TemporalLayerEntry() extends VisualSampleGroupEntry ('tscl')
{
   unsigned int(8)  temporalLayerId;

   unsigned int(2)  tlprofile_space;

   unsigned int(1)  tltier_flag;

   unsigned int(5)  tlprofile_idc;

   unsigned int(32) tlprofile_compatibility_flags;

   unsigned int(48) tlconstraint_indicator_flags;

   unsigned int(8)  tllevel_idc;

   unsigned int(16) tlMaxBitRate;

   unsigned int(16) tlAvgBitRate;

   unsigned int(8)  tlConstantFrameRate;

   unsigned int(16) tlAvgFrameRate;
}
```

8.4.5.2.3 Semantics

`temporalLayerId` gives the ID of this temporal layer. For all samples that are members of this sample group, the VCL NAL units shall have TemporalId, as defined in ISO/IEC 23008-2, equal to temporalLayerId.

`tlprofile_space`, `tltier_flag`, `tlprofile_idc`, `tlprofile_compatibility_flags`, `tlconstraint_indicator_flags`, and `tllevel_idc` contain the values of general_profile_space, general_tier_flag, general_profile_idc, general_profile_compatibility_flag[i] for i from 0 to 31, inclusive, the 6 bytes starting with the byte containing the general_progressive_source_flag, and general_level_idc, respectively, for the representation of the temporal layer identified by `temporalLayerId`.

`tlMaxBitrate` gives the maximum rate in 1000 bits per second over any window of one second, for the representation of the temporal layer identified by `temporalLayerId`.

`tlAvgBitRate` gives the average bit rate in units of 1000 bits per second, for the representation of the temporal layer identified by `temporalLayerId`.

`tlConstantFrameRate` equal to 1 indicates that the representation of the temporal layer identified by `temporalLayerId` is of constant frame rate. Value zero indicates that the representation of the temporal layer identified by `temporalLayerId` may or may not be of constant frame rate.

tlAvgFrameRate gives the average frame rate in units of frames/(256 seconds), for the representation of the temporal layer identified by temporalLayerId. Value 0 indicates an unspecified average frame rate.

8.4.6 Temporal sub-layer access sample grouping

8.4.6.1 Overview

An HEVC video track may contain zero or one instance of a SampleToGroupBox with a grouping_type 'tsas'. This SampleToGroupBox instance represents the marking of samples as temporal sub-layer access points. An accompanying instance of the SampleGroupDescriptionBox with the same grouping type shall be present.

8.4.6.2 Temporal sub-layer access sample group entry

8.4.6.2.1 Overview

Group Types: 'tsas'

Container: Sample Group Description Box ('sgpd')

Mandatory: No

Quantity: Zero or one

This sample group is used to mark temporal sub-layer access (TSA) samples. The grouping_type_parameter is not defined for the SampleToGroupBox with grouping type 'tsas'.

8.4.6.2.2 Syntax

```
class TemporalSubLayerEntry() extends VisualSampleGroupEntry ('tsas')

{

}
```

8.4.7 Step-wise temporal layer access sample grouping

8.4.7.1 Overview

An HEVC video track may contain zero or one instance of a SampleToGroupBox with a grouping_type 'stsa'. This SampleToGroupBox instance represents the marking of samples as step-wise temporal sub-layer access points. An accompanying instance of the SampleGroupDescriptionBox with the same grouping type shall be present.

When temporalIdNested in the applicable sample entry is equal to 1, the quantity of step-wise temporal sub-layer access sample group entry shall be zero.

8.4.7.2 Step-wise temporal layer sample group entry

8.4.7.2.1 Definition

Group Types: `'stsa'`

Container: Sample Group Description Box (`'sgpd'`)

Mandatory: No

Quantity: Zero or one

This sample group is used to mark step-wise temporal sub-layer access (STSA) samples. The `grouping_type_parameter` is not defined for the `SampleToGroupBox` with grouping type `'stsa'`.

8.4.7.2.2 Syntax

```
class StepwiseTemporalLayerEntry() extends VisualSampleGroupEntry ('stsa')

{

}
```

8.4.8 Definition of a sub-sample for HEVC

For the use of the `SubSampleInformationBox` (ISO/IEC 14496-12:2015, 8.7.7) in an HEVC stream, a sub-sample is defined on the basis of the value of the `flags` field of the sub-sample information box as specified below. The presence of this box is optional; however, if present in a track containing HEVC data, the '`codec_specific_parameters`' field in the box shall have the semantics defined here.

`flags` specifies the type of sub-sample information given in this box as follows:

— 0: NAL-unit-based sub-samples. A sub-sample contains one or more contiguous NAL units.

— 1: Decoding-unit-based sub-samples. A sub-sample contains exactly one decoding unit.

— 2: Tile-based sub-samples. A sub-sample either contains one tile and the associated non-VCL NAL units, if any, of the VCL NAL unit(s) containing the tile, or contains one or more non-VCL NAL units.

   NOTE 1   Tile-based sub-samples can be used to address individual tiles within a slice, whereas tile regions (as defined in subclause 9.6.4) can only be used for a set of tiles that are enclosed in one or more complete slice segments.

— 3: CTU-row-based sub-samples. A sub-sample either contains one CTU row within a slice and the associated non-VCL NAL units, if any, of the VCL NAL unit(s) containing the CTU row or contains one or more non-VCL NAL units. This type of sub-sample information shall not be used when entropy_coding_sync_enabled_flag is equal to 0.

— 4: Slice-based sub-samples. A sub-sample either contains one slice (where each slice may contain one or more slice segments, each of which is a NAL unit) and the associated non-VCL NAL units, if any, or contains one or more non-VCL NAL units.

— 5: Picture-based sub-samples. A sub-sample contains one coded picture and the associated non-VCL NAL units.

Other values of `flags` are reserved.

The subsample_priority field shall be set to a value in accordance with the specification of this field in ISO/IEC 14496-12.

The discardable field shall be set to 1 only if this sample is still decodable if this sub-sample is discarded (e.g. the sub-sample consists of an SEI NAL unit).

When the first byte of a NAL unit is included in a sub-sample, the preceding length field shall also be included in the same sub-sample.

The codec_specific_parameters field of the SubSampleInformationBox is defined for HEVC as follows:

```
if (flags == 0) {

    unsigned int(1) SubLayerRefNalUnitFlag;

    unsigned int(1) RapNalUnitFlag;

    unsigned int(1) VclNalUnitFlag;

    bit(29) reserved = 0;

} else if (flags == 1)

    bit(32) reserved = 0;

else if (flags == 2) {

    unsigned int(2) vcl_idc;

    bit(2) reserved = 0;

    unsigned int(4) log2_min_luma_ctb;

    unsigned int(12) ctb_x;

    unsigned int(12) ctb_y;

} else if (flags == 3 || flags == 4) {

    unsigned int(2) vcl_idc;

    bit(30) reserved = 0;

} else if(flags == 5) {

    unsigned int(1) DiscardableFlag;

    unsigned int(6) VclNalUnitType;

    unsigned int(6) LayerId;

    unsigned int(3) TempId;

    unsigned int(1) NoInterLayerPredFlag;

    unsigned int(1) SubLayerRefNalUnitFlag;

    bit(14) reserved = 0;

}
```

SubLayerRefNalUnitFlag equal to 0 indicates that all NAL units in the sub-sample are VCL NAL units of a sub-layer non-reference picture as specified in ISO/IEC 23008-2. Value 1 indicates that all NAL units in the sub-sample are VCL NAL units of a sub-layer reference picture as specified in ISO/IEC 23008-2.

RapNalUnitFlag equal to 0 indicates that none of the NAL units in the sub-sample has nal_unit_type equal to IDR_W_RADL, IDR_N_LP, CRA_NUT, BLA_W_LP, BLA_W_RADL, BLA_N_LP, RSV_IRAP_VCL22, or RSV_IRAP_VCL23 as specified in ISO/IEC 23008-2. Value 1 indicates that all NAL units in the sub-sample

have nal_unit_type equal to `IDR_W_RADL`, `IDR_N_LP`, `CRA_NUT`, `BLA_W_LP`, `BLA_W_RADL`, `BLA_N_LP`, `RSV_IRAP_VCL22`, or `RSV_IRAP_VCL23` as specified in ISO/IEC 23008-2.

`VclNalUnitFlag` equal to 0 indicates that all NAL units in the sub-sample are non-VCL NAL units. Value 1 indicates that all NAL units in the sub-sample are VCL NAL units.

`vcl_idc` indicates whether the sub-sample contains Video Coding Layer (VCL) data, non-VCL data, or both, as follows:

— 0: the sub-sample contains VCL data and does not contain non-VCL data

— 1: the sub-sample contains no VCL data and contains non-VCL data

— 2: the sub-sample may contain both VCL and non-VCL data, which shall be associated with each other. For example, a sub-sample may contain a decoding unit information SEI message followed by the set of NAL units associated with the SEI message.

— 3: reserved

`log2_min_luma_ctb` indicates the unit of `ctb_x` and `ctb_y`, specified as follows:

— 0: 8 luma samples

— 1: 16 luma samples

— 2: 32 luma samples

— 3: 64 luma samples

`ctb_x` specifies the 0-based coordinate of the right-most luma samples of the tile associated with the sub-sample when `flags` is equal to 2 and `vcl_idc` is equal to 1 or 2, in units derived from `log2_min_luma_ctb` as specified above.

`ctb_y` specifies the 0-based coordinate the bottom-most luma samples of the tile associated with the sub-sample when `flags` is equal to 2 and `vcl_idc` is equal to 1 or 2, in units derived from `log2_min_luma_ctb` as specified above.

`DiscardableFlag` indicates the discardable_flag value of the VCL NAL units in the sub-sample. All the VCL NAL units in the sub-sample shall have the same discardable_flag value.

NOTE 2   This is not the same definition as the `discardable` field in the `SubSampleInformationBox`.

`VclNalUnitType` indicates the nuh_unit_type value of the VCL NAL units in the sub-sample. All the VCL NAL units in the sub-sample shall have the same nuh_unit_type value.

`LayerId` indicates the nuh_layer_id value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same nuh_layer_id value.

`TempId` indicates the TemporalId value of the NAL units in the sub-sample. All the NAL units in the sub-sample shall have the same TemporalId value.

`NoInterLayerPredFlag` indicates the value of the inter_layer_pred_enabled_flag of the VCL NAL units in the sub-sample. All the VCL NAL units in the sub-sample shall have the same value of inter_layer_pred_enabled_flag.

8.4.9   Handling non-output samples

HEVC allows for file format samples that are used only for reference and not output (e.g. a non-displayed reference picture in video). When any such non-output sample is present in a track, the file shall be constrained as follows:

1)   A non-output sample shall be given a composition time that is outside the time-range of the samples that are output.

2) An edit list shall be used to exclude the composition times of the non-output samples.

3) When the track includes a CompositionOffsetBox ('ctts'),

 a. version 1 of the CompositionOffsetBox shall be used,

 b. the value of `sample_offset` shall be set equal to -2^{31} for each non-output sample,

 c. the CompositionToDecodeBox ('cslg') should be contained in the SampleTableBox ('stbl') of the track, and

 d. when the CompositionToDecodeBox is present for the track, the value of `leastDecodeToDisplayDelta` field in the box shall be equal to the smallest composition offset in the CompositionOffsetBox excluding the `sample_offset` values for non-output samples.

 NOTE   Thus, `leastDecodeToDisplayDelta` is greater than -2^{31}.

9 Layered HEVC elementary stream and sample definitions

9.1 Overview

This clause specifies the storage format of coded video data of layered HEVC (L-HEVC), which includes all the layered HEVC extensions that use the same layered design specified in ISO/IEC 23008-2:2017, Annex F. These layered HEVC extensions include SHVC, MV-HEVC, and 3D-HEVC. This clause also specifies the storage of temporal sub-layers of a layer in more than one track. This clause extends the definitions of the storage format of HEVC in Clause 8.

The file format as defined in this clause and Annex A uses the existing capabilities of the ISO base media file format and the plain HEVC file format (i.e. the file format specified in Clause 8). The following structures or extensions, among others, to support L-HEVC-specific features are used:

a) Aggregator:

 a structure to enable efficient scalable grouping of NAL units by changing irregular patterns of NAL units into regular patterns of aggregated data units.

 NOTE 1   When compared to Aggregators for AVC/SVC/MVC, the syntax and semantics of the NAL unit header syntax element in the L-HEVC aggregator have been modified and the scope of the aggregator has been constrained.

b) Extractor:

 a structure to enable efficient extraction of NAL units from other tracks than the one containing the media data.

 NOTE 2   When compared to Extractors for AVC/SVC/MVC, the syntax and semantics of the L-HEVC Extractors have been modified.

c) HEVC compatibility:

 a provision for storing an L-HEVC bitstream in an HEVC compatible manner, such that the HEVC compatible base layer can be used by any plain HEVC file format compliant reader. L-HEVC data can be stored with the base layer or in one or more dedicated tracks.

d) External codec compatibility:

 a provision for storing an L-HEVC bitstream in a manner that is compatible to an external codec (e.g., AVC) that was used for encoding of the externally provided base layer; this way the base layer that is compatible to an external codec can be used by any file format reader compliant to that external codec (e.g., plain AVC file format compliant reader).

9.2 Overview of L-HEVC storage

The support for L-HEVC includes a number of tools, and there are various 'models' of how they might be used. In particular, an L-HEVC stream can be placed in tracks in a number of ways, among which are the following:

1. all the layers in one track;

2. each layer or sub-layer in its own track;

3. the expected operating points each in a track (e.g. the HEVC base, a stereo pair, a multiview scene).

The L-HEVC file format allows storage of one or more layers into a track. Storage of multiple layers per track can be used. For example, when a content provider wants to provide a multi-layer bitstream that is not intended for subsetting, or when the bitstream has been created for a few pre-defined sets of output layers where each layer corresponds to a view (such as 1, 2, 5, or 9 views), tracks can be created accordingly.

When an L-HEVC bitstream is represented by multiple tracks and a player uses an operating point for which the layers are stored in multiple tracks, the player must reconstruct L-HEVC access units before passing them to the L-HEVC decoder. An L-HEVC operating point may be explicitly represented by a track, i.e., each sample in the track contains an access unit, where some or all NAL units of the access unit may be contained in or referred to by extractors and aggregators. If the number of operating points is large, it may be space-consuming and impractical to create a track for each operating point. In such a case, L-HEVC access units are reconstructed as specified in subclause 9.5.2.2.

The storage of L-HEVC bit streams is supported by structures such as the sample entry, Operating Points Information ('oinf') sample group, and Layer Information ('linf') sample group. The structures within a sample entry provide information for the decoding or use of the samples, in this case coded video information, that are associated with that sample entry. The Operating Points Information sample group records information about operating points such as the layers and sub-layers that constitute the operating point, dependencies (if any) between them, the profile, level, and tier parameter of the operating point, and other such operating point relevant information. The layer information sample group lists all the layers and sub-layers carried in the samples of the track. The information in these sample groups, combined with using track references to find tracks, is sufficient for a reader to choose an operating point in accordance with its capabilities, identify the tracks that contain the relevant layers and sub-layers needed to decode the chosen operating point, and efficiently extract them.

When the samples of a track contain temporal sub-layers of an HEVC base layer but do not contain the temporal sub-layer with TemporalId equal to 0 of an HEVC base layer, an 'hvc2', or 'hev2' sample entry shall be used. When the samples of a track contain an HEVC compatible base layer or a temporal subset of an HEVC base layer including a sub-layer with TemporalId equal to 0, an 'hvc1', 'hev1', 'hvc2', or 'hev2' sample entry shall be used.

NOTE   When a sample entry type 'hvc2' or 'hev2' is used in a track containing the base layer, parsers complying only with non-layered HEVC storage specified in Clause 8 are not able to process the track.

The L-HEVC file format supports a hybrid codec configuration where the base layer is coded using an external video codec (e.g., AVC). The base layer, if coded using an external codec specification (e.g., ISO/IEC 14496-10), shall be stored according to the external codec file format (e.g., the AVC file format as specified in subclause 4.11).

9.3 L-HEVC elementary stream structure

L-HEVC streams are stored in accordance with subclause 8.2, with the exception that an L-HEVC stream may be represented by more than one video track in a file, and with the following definition of an L-HEVC video elementary stream:

— **An L-HEVC video elementary stream** contains all video coding related NAL units (i.e. those NAL units containing video data or signalling video structure, possibly after resolution of extractors

and or aggregators) and may contain non-video coding related NAL units such as SEI messages and access unit delimiter NAL units. Aggregators and Extractors, when present, shall not be directly output by file parsers.

9.4 Sample and configuration definition

9.4.1 Overview

An L-HEVC sample consists of the picture units that are within an L-HEVC access unit and that are represented by the track. A sample in an `'hvc1'` or `'hev1'` track with the L-HEVC configuration is an L-HEVC sample, while a sample in an `'hvc1'` or `'hev1'` track without the L-HEVC configuration is not an L-HEVC sample. Throughout Clause 9, unless mentioned otherwise, an access unit refers to an L-HEVC access unit and is as defined in ISO/IEC 23008-2:2017, Annex F.

9.4.2 Canonical order and restrictions

The following restrictions apply to L-HEVC data in addition to the requirements in subclause 8.3.2.

— **VCL NAL units**: All VCL NAL units within one access unit and belonging to the layers represented by a track shall be contained in the sample and the sample shall not contain NAL units of any other access unit. After resolution of extractors and aggregators, when present, an L-HEVC sample shall contain an integer number of picture units. The picture units in an L-HEVC sample shall be stored in their decoding order.

— **Aggregators and extractors**: Each aggregator is allowed to aggregate NAL units belonging to one picture unit only. The order of all NAL units included in an aggregator or in a resolved extractor is exactly the decoding order as if these NAL units were present in a sample not containing aggregators or extractors. After processing the aggregator or the extractor, all NAL units must be in valid decoding order as specified in ISO/IEC 23008-2.

— **Carriage of external-codec coded base layer track:** An external-codec (e.g., AVC) coded base layer shall always be carried in a track of its own, and shall consist of samples as specified in the external codec file format (e.g., the AVC file format as specified in subclause 4.11).

9.4.3 Decoder configuration record

When the decoder configuration record defined in subclause 8.3.3.1 is used for a stream that can be interpreted as either an L-HEVC or HEVC stream, the HEVC decoder configuration record shall apply to the HEVC compatible base layer, and should contain only parameter sets needed for decoding the HEVC base layer.

The syntax of LHEVCDecoderConfigurationRecord is as follows:

```
aligned(8) class LHEVCDecoderConfigurationRecord {

    unsigned int(8) configurationVersion = 1;

    bit(4) reserved = '1111'b;

    unsigned int(12) min_spatial_segmentation_idc;

    bit(6) reserved = '111111'b;

    unsigned int(2) parallelismType;

    bit(2) reserved = '11'b;

    unsigned int(3) numTemporalLayers;

    unsigned int(1) temporalIdNested;
```

```
    unsigned int(2) lengthSizeMinusOne;

    unsigned int(8) numOfArrays;

    for (j=0; j < numOfArrays; j++) {

        unsigned int(1) array_completeness;

        bit(1) reserved = 0;

        unsigned int(6) NAL_unit_type;

        unsigned int(16) numNalus;

        for (i=0; i< numNalus; i++) {

            unsigned int(16) nalUnitLength;

            bit(8*nalUnitLength) nalUnit;

        }

    }

}
```

The semantics of the fields that are common to `LHEVCDecoderConfigurationRecord` and `HEVCDecoderConfigurationRecord` remain unchanged. If the track containing `LHEVCDecoderConfigurationRecord` has a `'scal'` track reference, the semantics of the fields apply to the output bitstream resulting from the explicit reconstruction process. Otherwise, if there is at least one operating point for which the highest output layer is natively present in the track, the semantics of the fields apply to all (implicitly reconstructed) output bitstreams of the operating points for which the highest sub-layer of the highest output layer is natively present in the track. Otherwise, the semantics of the fields apply to all sub-layers natively present in the track.

NOTE   For each auxiliary picture layer included in the track, it is recommended to include, within `nalUnit`, an SEI NAL unit containing a declarative SEI message, such as the depth representation information SEI message for depth auxiliary picture layers, specifying characteristics of the auxiliary picture layer.

9.5   Derivation from the ISO base media file format and the HEVC file format (Clause 8)

9.5.1   L-HEVC track structure

An L-HEVC stream is represented by one or more video tracks in a file. Each track represents one or more layers or sub-layers of the coded bitstream.

There is a minimal set of one or more tracks that, when taken together, contain the complete set of encoded information. This group of tracks that forms the complete encoded information is called the "complete subset". The complete encoded information can be retained when the tracks included in the "complete subset" are retained; all other tracks shall represent subsets or copies (through extractors) of the complete subset.

All tracks of the complete subset shall be indicated with a track group of type `'cstg'` (complete subset track grouping).

For each valid pair of nuh_layer_id and TemporalId among all the coded pictures of an L-HEVC bitstream, the complete subset shall include one and only one track in which the `'linf'` sample group indicates that the sub-layer identified by nuh_layer_id and TemporalId is natively present.

NOTE 1   Consequently, a file reader can extract a sub-bitstream of any operating point without the need to parse Aggregators and Extractors.

For an L-HEVC bitstream with an external base layer (i.e., when an active VPS of the bitstream has vps_base_layer_internal_flag equal to 0 and vps_base_layer_available_flag equal to 1), the base layer is always assigned one track of its own. A track containing an external base layer for an L-HEVC bitstream shall not be a part of the 'cstg' track group of the tracks of that L-HEVC bitstream.

If an L-HEVC bitstream has an external base layer, the track containing the external base layer is nominated as the base track. Otherwise, the track with the lowest temporal sub-layer (the VCL NAL units of which have TemporalId equal to 0) of the base layer natively present is nominated as the base track.

All the other tracks that are part of the same stream shall be linked to the base track by means of a track reference of type 'sbas'. All the tracks sharing the same base track shall share the same timescale as the scalable base track.

Tracks that contain extractors shall be linked to the tracks from which they extract data using the 'scal' track reference type.

NOTE 2   If a track containing parts of an L-HEVC bitstream is removed from a file, tracks that contain 'scal' and 'sbas' track references to the removed track should also be removed.

When the base layer is coded using AVC, the base layer track shall be constructed according to clause 4.11 without using separate parameter set tracks.

9.5.2   Data sharing and reconstruction of an L-HEVC bitstream

9.5.2.1   General

NOTE 1   When only the entire base layer carried in the base track is played or when only certain sub-layers, carried in the base track, of the base layer are played, and the base track natively contains all the required sub-layers of the base layer as identified by the HEVCDecoderConfigurationRecord of the base track, the reconstruction processes of this clause are unnecessary.

Different tracks may logically share data through the use of extractors (defined in clause A.7).

In order to reconstruct an access unit from samples of multiple tracks carrying an L-HEVC bitstream, the target output layers and the operating point they belong to may need to be determined first.

NOTE 2   Players can conclude the layers that are required for decoding the determined target output layers from operating point list included in the Operating Points Information sample group. Tracks that carry the relevant layers for an operating point can be obtained by following the 'oref' track references and information in the layer information sample group.

If several tracks contain data for the access unit, the alignment of respective samples in tracks is performed based on the sample decoding times, i.e. using the time-to-sample table only without considering edit lists.

When an L-HEVC stream is represented by multiple tracks, the decoding times of the samples shall be such that if the tracks were combined into a single stream ordered by increasing decoding time, the access unit order would be correct as specified in ISO/IEC 23008-2.

If the track containing the highest sub-layer of the highest layer of a desired operating point has 'scal' track references, all access units of the track for that operating point are reconstructed by solving the extractors and aggregators in the samples of that track, and the result is the bitstream of the operating point; this reconstruction process is referred to as explicit reconstruction.

Otherwise, a sequence of access units is reconstructed from the respective samples in the required tracks according to the implicit reconstruction process as described in subclause 9.5.2.2.

9.5.2.2   Implicit reconstruction

In the implicit reconstruction, the required tracks are selected based on the layers they carry and their reference layers as indicated by the Operating Points Information and Layer Information sample groups.

When reconstructing a bitstream containing a sub-layer for which the VCL NAL units have TemporalId greater than 0, all lower sub-layers (i.e., those for which the VCL NAL units have smaller TemporalId) within the same layer are also included in the resulting bitstream and the required tracks are selected accordingly.

Each access unit is reconstructed as described below, the reconstructed access units are placed into the L-HEVC bitstream in increasing order of decoding time, and the duplicates of end of bitstream (EOB) and end of sequence (EOS) NAL units are removed from the L-HEVC bitstream, as described further below.

Aggregators, when present, are resolved, while extractors, when present (natively present or resolved from aggregators), are discarded without being resolved, and the resultant NAL units are arranged in an order conforming to ISO/IEC 23008-2.

When reconstructing an access unit, picture units (as specified in ISO/IEC 23008-2) from samples having the same decoding time are placed into the access unit in increasing order of the nuh_layer_id value. The reconstructed access units are placed into the output bitstream in increasing order of the decoding time.

For a particular access unit for which the NAL units are stored in multiple tracks, there may be more than one of the tracks containing an EOB NAL unit in the respective samples. In this case, only one of the EOB NAL units shall be kept in the final reconstructed access unit, placed at the end of the access unit; other EOB NAL units are discarded.

For access units that are within the same coded video sequence of an HEVC or L-HEVC bitstream and that belong to different sub-layers stored in multiple tracks, there may be more than one of the tracks containing an EOS NAL unit in the respective samples. In this case, only one of the EOS NAL units shall be kept in the last of these access units (the one with the greatest decoding time) in the final reconstructed bitstream, placed after all NAL units, except the EOB NAL unit (when present), of the last of these access units, and other EOS NAL units are discarded. Similarly, there may be more than one of such tracks containing an EOB NAL unit in the respective samples. In this case, only one of the EOB NAL units shall be kept in the final reconstructed bitstream, placed at the end of the last of these access units, and other EOB NAL units are discarded.

Since a particular layer or sub-layer may be represented by more than one track, when figuring out the required tracks for an operating point, a selection may need to be made among the set of tracks that all carry the particular layer or sub-layer.

The final required tracks, after selection among the tracks carrying a same layer or sub-layer, may still collectively carry some layers or sub-layers that do not belong to the target operating point. The reconstructed bitstream for the target operating point should not contain the layers or sub-layers that are carried in the final required tracks but do not belong to the target operating point.

NOTE   Some L-HEVC decoder implementations take as input a bitstream as well as the target output layer set index and the highest TemporalId value of the target operating point, which correspond to the TargetOlsIdx and HighestTid variables in ISO/IEC 23008-2:2017, F.8, respectively. In this case the bitstream reconstructed by the file parser can contain layers or sub-layers that do not belong to the target operating point, because these L-HEVC decoders are capable of removing the layers not included in the target output layer set and the sub-layers beyond the highest TemporalId value. Some other L-HEVC decoder implementations input a bitstream that is required not to contain any other layers and sub-layers than those included in the target operating point. In this case the file parser needs to ensure that the reconstructed bitstream does not contain any other layers and sub-layers than those included in the target operating point.

9.5.3   L-HEVC video stream definition

9.5.3.1   Sample entry name and format

9.5.3.1.1   Definition

Sample Entry and Box Types: 'hvc2', 'hev2', 'lhv1', 'lhe1', 'lhvC'

Container: Sample Description Box ('stsd')

Mandatory: An 'hvc1', 'hev1', 'hvc2', 'hev2', 'lhv1', or 'lhe1' sample entry is mandatory

Quantity: One or more sample entries may be present

When the sample entry name is 'lhv1', the default and mandatory value of array_completeness is 1 for arrays of all types of parameter sets, and 0 for all other arrays. When the sample entry name is 'lhe1', the default value of array_completeness is 0 for all arrays.

When the sample entry name is 'hev2' and the sample entry contains the HEVC configuration only, the same constraints as specified in subclause 8.4.3 for the sample entry name 'hev1' apply.

When the sample entry name is 'lhe1', or when the sample entry name is 'hev1' or 'hev2' and the sample entry contains both HEVC and L-HEVC configurations, the following applies:

NOTE 1   The constraints below impose restrictions on placement of out-of-band parameter sets (in sample entries) and in-band parameter sets (in samples), in order to enable convenient random access from access units containing IRAP pictures at least in some layers. With these constraints, a file reader that initializes with the sample entries and rolls forward from an access unit wherein all pictures are IRAP pictures will have all the parameter sets it needs.

— For any particular sample in a particular track, the temporally collocated sample in another track is defined as the one with the same decoding time as that of this particular sample.

— For an IRAP picture of a given sample, track and layer, each parameter set needed for decoding the IRAP picture shall be included in one of the following:

   a.   the sample entry that applies to the given sample in the given track

   b.   when implicit reconstruction is used, the sample entry of the initial sample of a track carrying a reference layer of the given layer, where the initial sample is either the given sample's temporally collocated sample, when the temporally collocated sample contains an IRAP picture of the reference layer, or the previous sample that contains an IRAP picture of the reference layer

   c.   the given sample itself, possibly by using extractors

   d.   when present, any temporally collocated sample of the tracks carrying reference layers of the given layer, possibly by using extractors

— For a non-IRAP picture of a given sample, track and layer, each parameter set needed for decoding that picture shall be included in one of the following:

   a.   the sample entry that applies to the given sample in the given track

   b.   when implicit reconstruction is used, the sample entry of the initial sample of a track carrying a reference layer of the given layer, where the initial sample is either the given sample's temporally collocated sample, when the temporally collocated sample contains an IRAP picture of the reference layer, or the previous sample that contains an IRAP picture of the reference layer

   c.   any of the samples in the given track since the previous sample containing an IRAP picture in the given layer to the given sample itself, inclusive, possibly by using extractors

d. when present, any of the samples in a track carrying a reference layer of the given layer since the temporally collocated sample of the previous sample containing an IRAP picture in the given layer to the temporally collocated sample of the given sample, inclusive, possibly by using extractors

The parameter sets needed for decoding a picture in a sample of a track using extractors shall only be located in media samples (of that track or of dependent tracks through extractors), or in the sample description entry associated with that sample, and parameter sets carried in sample entries of other tracks are ignored in this case. This implies that when extractors are used, some parameter sets may be duplicated in the sample description of tracks with extractors and in the other tracks depended on this track. When implicit reconstruction is used, the sample entries of a track shall only contain the parameter sets used in that track and shall not contain the parameter sets used in dependent or depended on tracks. These parameter sets may be stored in the sample or in the sample description associated to the sample, according to the sample entry type used.

When present, parameter sets in the sample entry of an HEVC/L-HEVC track and, in case implicit reconstruction is used, in the sample entries of the tracks depended on, shall be output in the reconstructed bitstream in increasing layer order, and before any parameter sets present in the samples.

In case two parameter sets with different content but using the same ID are present, it may not be possible to use a single sample entry of type `'hvc1'`, `'hvc2'` or `'lhv1'`; file packagers should create either different sample entries of type `'hvc1'`, `'hvc2'` or `'lhv1'`, or use in-band parameter sets through `'hev1'`, `'hev2'` or `'lhe1'` sample entries.

NOTE 2   It can happen, in files mixing both implicit reconstructions and extractors, that gathering parameter sets from sample entries result in duplicates; additionally, presence of such duplicates are conformant with ISO/IEC 23008-2, a file reader can discard them.

NOTE 3   When an extractor track is used in implicit reconstruction, file readers don't need to look at the sample entries of tracks the extractor track depends on for their parameter sets, since these parameter sets are duplicated in the extractor track's sample entry.

For an HEVC or L-HEVC bitstream carried in more than one track, when the sample entry name of the base track is `'hvc1'` or `'hvc2'`, the sample entry name of other tracks carrying the same bitstream shall be `'hvc2'` or `'lhv1'`, and when the sample entry name of the base track is `'hev1'` or `'hev2'`, the sample entry name of other tracks carrying the same bitstream shall be `'hev2'` or `'lhe1'`.

For an L-HEVC bitstream whose base layer is an AVC bitstream, when the sample entry name of the base track is `'avc1'` or `'avc2'`, the sample entry name of the other tracks carrying the associated L-HEVC bitstream shall be `'hvc2'` or `'lhv1'`, and when the sample entry name of the base track is `'avc3'` or `'avc4'`, the sample entry name of the other HEVC tracks carrying the associated L-HEVC bitstream shall be `'hev2'` or `'lhe1'`.

NOTE 4   The above constraints mandate convenient random access (see NOTE 1 above in this clause) to be enabled and indicated for either all tracks carrying an HEVC or L-HEVC bitstream or none of the tracks.

When the samples of a track contain a temporal subset of an HEVC base layer, the value of `numTemporalLayers` field in the `HEVCConfigurationBox` shall match the number of temporal sub-layers present in the track, natively or through extraction. Here, the entry shall contain an `HEVCConfigurationBox`, possibly followed by an `LHEVCConfigurationBox` as defined below. The `HEVCConfigurationBox` documents the Profile, Tier, Level, and possibly also parameter sets of the HEVC compatible base layer as defined by the `HEVCDecoderConfigurationRecord`. The `LHEVCConfigurationBox` possibly documents parameter sets of the L-HEVC compatible enhancement layers as defined by the `LHEVCDecoderConfigurationRecord`, stored in the `LHEVCConfigurationBox`.

If the samples of a track do not contain, neither natively nor through extraction, an HEVC base layer, then the sample entry type `'lhv1'` or `'lhe1'` shall be used and the sample entry shall contain an `LHEVCConfigurationBox`, as defined below. This includes an `LHEVCDecoderConfigurationRecord`, as defined in this document.

The `lengthSizeMinusOne` field in the L-HEVC and HEVC configurations in any given sample entry of L-HEVC and HEVC tracks sharing the same base track shall have the same value.

NOTE 5   When the base layer is coded by AVC, the `lengthSizeMinusOne` field for the L-HEVC tracks sharing the same base track shall still have the same value, but that value does not have to be equal to the `lengthSizeMinusOne` field for the AVC track.

NOTE 6   When HEVC compatibility is indicated, it can be necessary to indicate a high enough level for the HEVC base layer to accommodate the bit rate of the entire bitstream, because all the NAL units are considered as included in the HEVC base layer and hence can be fed to the decoder, which is expected to discard those NAL unit it does not recognize. This case happens when the `'hvc1'`, `'hev1'`, `'hvc2'`, or `'hev2'` sample entry is used and both HEVC and L-HEVC configurations are present. However, when only an HEVC and L-HEVC configuration is present, on the contrary, the optimal level for the HEVC track is the one that only accommodates the bit rate of the base layer bitstream. In this case, caution should be taken that the profile, tier and level (PTL) in the VPS base (i.e. the first PTL structure in a VPS) might be "greater" than the PTL of the base layer that should be included in the `HEVCDecoderConfigurationRecord`. Rather, the PTL information in the first PTL structure in the VPS extension is what should be included in `HEVCDecoderConfigurationRecord`.

An `LHEVCConfigurationBox` may be present in an `'hvc1'`, `'hev1'`, `'hvc2'`, or `'hev2'` sample entry. In this case the `HEVCLHVCSampleEntry` definition below applies.

Table 11 shows for a video track all the possible uses of sample entries, configurations and the L-HEVC tools:

Table 11 — Use of sample entries for HEVC and L-HEVC tracks

sample entry name	with configuration records	meaning
`'hvc1'` or `'hev1'`	HEVC Configuration Only	A plain HEVC track without NAL units with nuh_layer_id greater than 0; Extractors and aggregators shall not be present.
`'hvc1'` or `'hev1'`	HEVC and L-HEVC Configurations	An L-HEVC track with both NAL units with nuh_layer_id equal to 0 and NAL units with nuh_layer_id greater than 0; Extractors and aggregators shall not be present.
`'hvc2'` or `'hev2'`	HEVC Configuration Only	A plain HEVC track without NAL units with nuh_layer_id greater than 0; Extractors may be present and used to reference NAL units; Aggregators may be present to contain and reference NAL units.
`'hvc2'` or `'hev2'`	HEVC and L-HEVC Configurations	An L-HEVC track with both NAL units with nuh_layer_id equal to 0 and NAL units with nuh_layer_id greater than 0; Extractors and aggregators may be present; Extractors may reference any NAL units; Aggregators may both contain and reference any NAL units.
`'lhv1'`, `'lhe1'`	L-HEVC Configuration Only	An L-HEVC track with NAL units with nuh_layer_id greater than 0 and without NAL units with nuh_layer_id equal to 0; Extractors shall not be present; Aggregators may be present to contain and reference NAL units.

When an `'hvc2'`, `'hev2'`, `'lhv1'`, or `'lhe1'` track does not contain extractors and does not contain VCL NAL units with TemporalId equal to 0, the track shall contain VCL NAL units with exactly one nuh_layer_id value.

9.5.3.1.2   Syntax

```
class LHEVCConfigurationBox extends Box('lhvC') {

    LHEVCDecoderConfigurationRecord() LHEVCConfig;

}

class HEVCLHVCSampleEntry() extends HEVCSampleEntry() {
```

```
    LHEVCConfigurationBox      lhvcconfig;

}

// Use this if track is not HEVC compatible

class LHEVCSampleEntry() extends VisualSampleEntry ('lhv1', or 'lhe1') {

    LHEVCConfigurationBox      lhvcconfig;

    MPEG4ExtensionDescriptorsBox (); // optional

}
```

9.5.3.1.3   Semantics

When the sample entry is `'lhv1'` or `'lhe1'`, `Compressorname` in the base class `VisualSampleEntry` indicates the name of the compressor used with the value "\014LHEVC Coding" being recommended (\014 is 12, the length of the string "LHEVC Coding" in bytes). When the sample entry is `'hvc2'` or `'hev2'`, `Compressorname` in the base class `VisualSampleEntry` indicates the name of the compressor used with the value "\013HEVC Coding" being recommended (\013 is 11, the length of the string "HEVC Coding" in bytes).

9.5.4   L-HEVC visual width and height

The `width` and `height` documented in the `VisualSampleEntry` shall be set as follows:

— If the sample entry type is `'hvc1'` or `'hev1'`, the `width` and `height` documented in the `VisualSampleEntry` shall be set according to subclause 4.6 using only the base layer information.

— Otherwise, if the sample entry type is `'hvc2'` or `'hev2'` and the track does not contain an L-HEVC configuration record, the `width` and `height` documented in the `VisualSampleEntry` shall be set according to the width and height of the base layer of the bitstream resulting either by resolving the extractors, when the track contains extractors, or by implicit reconstruction, when the track does not contain extractors.

— Otherwise (the sample entry type is `'hvc2'` or `'hev2'` and the track contains an L-HEVC configuration record or the sample entry type is `'lhe1'` or `'lhv1'`), `width` and `height` shall be the maximum cropped frame dimensions of the decoded pictures, within the scope of the sample entry, of any layer in the track that is marked as an output layer of any output layer set.

9.5.5   Sync sample

Sync samples in a track of a particular sample entry type shall conform to the following constraints:

— If the sample entry type is `'hvc1'` or `'hev1'` and the track does not contain an L-HEVC configuration record, the specifications of subclause 8.4.3 apply.

— Otherwise, if the sample entry type is `'hvc1'` or `'hev1'` and the track contains an L-HEVC configuration record, each sync sample of the track shall contain an IRAP picture in all layers represented by the track.

— Otherwise, if the sample entry type is `'hvc2'` or `'hev2'` and the track contains extractors, each sync sample of the track shall contain an IRAP picture in all layers represented by the track either natively or through extraction.

— Otherwise, if the sample entry type is `'hvc2'` or `'hev2'`, and the track does not contain extractors, contains a sub-layer with TemporalId equal to 0 for one or more layers, and does not contain an L-HEVC configuration record, each sync sample of the track shall contain an IRAP picture with nuh_layer_id equal to 0.

— Otherwise, if the sample entry type is `'hvc2'`, `'hev2'`, `'lhv1'`, or `'lhe1'` and the track does not contain extractors, contains a sub-layer with TemporalId equal to 0 for one or more layers, and contains an L-HEVC configuration record, each sync sample of the track shall contain an IRAP picture in all layers natively present in the track.

— Otherwise (the sample entry type is `'hvc2'`, `'hev2'`, `'lhv1'`, or `'lhe1'`, and the track does not contain extractors and does not contain any sub-layer with TemporalId equal to 0), each sync sample of the track shall contain a TSA picture with TemporalId equal to the lowest TemporalId present in the track.

NOTE 1   As can be observed from the list above, in some cases sync samples require the presence of an IRAP picture in more than one layer. Should layer-specific information be needed, the `'sap '` sample grouping can be used.

Sync samples are documented in the `SyncSampleBox` and, for movie fragments, with `sample_is_non_sync_sample` flag equal to 0, as specified in ISO/IEC 14496-12. Sync samples may be additionally documented by the stream access point `'sap '` sample group.

When a sync sample contains natively or through extraction a BLA or CRA picture in a layer that does not have an alternative output layer, as defined by HEVC, in an output layer set represented by the track, there shall be no RASL pictures associated with the BLA or CRA picture.

Previous editions of this document allowed the presence of RASL pictures associated with BLA and CRA pictures contained in sync samples, although this is not in conformance with ISO/IEC 14496-12. For environments where HEVC file processors need information that BLA or CRA pictures contained in sync samples are not associated with RASL pictures, the brand `'nras'`, as defined in Annex D.5, may be used to indicate compliance to this and later versions of this document.

NOTE 2   The presence of RASL pictures that would be needed for output is not strictly in conformance with the sync sample definition of ISO/IEC 14496-12 and hence it is recommended they not be present.

9.5.6   Independent and disposable samples box

If the `SampleDependencyTypeBox` is used in a track that is both HEVC and L-HEVC compatible, then care should be taken that the information provided by this box is true no matter what valid subset of the L-HEVC data (possibly only the HEVC data) is used. The 'unknown' values (value 0 of the fields `sample_depends_on`, `sample_is_depended_on`, and `sample_has_redundancy`) may be needed if the information varies.

9.5.7   Stream access point sample group

To provide information of all SAPs, the SAP sample group `'sap '` specified in ISO/IEC 14496-12 is used.

The semantics of `layer_id_method_idc` equal to 0 is specified in ISO/IEC 14496-12.

When layer_id_method_idc is equal to 0, a SAP is interpreted as follows:

— If the sample entry type is `'hvc2'`, `'hev2'`, `'lhv1'`, or `'lhe1'`, and the track does not contain extractors and does not contain any sub-layer with TemporalId equal to 0, a SAP specifies access to all the sub-layers present in the track.

— Otherwise, a SAP specifies access to all layers present natively or through extraction in the track.

NOTE   If the sample entry type is `'hvc2'`, `'hev2'`, `'lhv1'`, or `'lhe1'`, and the track does not contain extractors and does not contain any sub-layer with TemporalId equal to 0, a TSA picture with TemporalId equal to the lowest TemporalId present in the track serves as a SAP.

When `layer_id_method_idc` is equal to 1, each bit in the field `target_layers` represents a layer carried in the track. Since this field is only 28 bits in length, the indication of SAPs in a track is constrained to a maximum of 28 layers. Each bit of this field starting from the least significant bit (LSB) shall be mapped to the list of `layer_id` values signalled in the Layer Information sample group (`'linf'`) associated with that sample, in ascending order of `layer_id` values.

For example, if a track carries layer with `layer_id` values 4, 10, and 29, then the `layer_id` 4 is mapped to the least significant bit, the `layer_id` 10 is mapped to the second least significant bit, and the `layer_id` 29 maps to the third least significant bit. A value of one in the bit signals that in the sample the mapped layer has a picture that is a SAP. In the previous example Table 12 gives an example of layer specific SAP information.

Table 12 — Bit pattern for the target_layer field of the `'sap '` sample grouping for an example use case described above. (informative)

Bit pattern of target layers (LSB right most bit)	`layer_id` values (among 4, 10, 29) of SAPs
0000000000000000000000010	{10}
0000000000000000000000100	{29}
0000000000000000000000101	{29, 4}

When SAP sample groups are used, they shall be used on all tracks carrying an L-HEVC bitstream. When extractors are used in a track, the SAPs of those layers referred to by extractors are also documented. The SAP documentation shall be consistent for any given layer.

9.5.8   The `'roll'`, `'rap '`, `'sync'`, `'tsas'` and `'stsa'` sample groups

When included in the base track, the information provided by the `'roll'`, `'rap '`, and `'sync'` sample groups applies to the base layer only, not considering any other layers that are present in the samples.

When included in an `'hvc2'` or `'hev2'` track that is not the base track and contains extractors, the information provided by the `'roll'`, `'rap '`, and `'sync'` sample groups in `SampleToGroupBox`(es) with `version` equal to 0 applies to the track, after resolving extractors and aggregators contained in the track.

When included in an `'hvc2'`, `'hev2'`, `'lhv1'`, or `'lhe1'` track that is not the base track, does not contain extractors, and natively contains only one layer (including its lowest sub-layer), the information provided by the `'roll'`, `'rap '`, and `'sync'` sample groups in `SampleToGroupBox`(es) with `version` equal to 0 applies to that layer.

NOTE   When a `'roll'`, `'rap '`, or `'sync'` sample group concerns a predicted layer, it indicates characteristics that apply when all the reference layers of the predicted layer are available and decoded. The sample group can be used to initiate decoding of the predicted layer.

The presence of the `'sync'`, `'roll'`, and `'rap '` sample groups in `'hvc2'`, `'hev2'`, `'lhv1'`, or `'lhe1'` tracks natively containing more than one layer or natively containing one layer but not its lowest sub-layer is disallowed in this document, and parsers according to this document shall ignore `SampleGroupDescriptionBoxes` and `SampleToGroupBoxes` with `grouping_type` equal to `'sync'`, `'roll'`, or `'rap '` within such tracks.

For the `'tsas'` and `'stsa'` sample groups, the specifications in subclause 9.5.7 apply with `'sap '` being replaced by `'tsas'` and `'stsa'`, respectively, and "SAP" being replaced by "TSA point" and "STSA point", respectively.

9.5.9   Definition of a sub-sample for L-HEVC

The specifications of subclause 8.4.8 apply.

9.5.10   Handling non-output samples

What is specified in subclause 8.4.9 is not always applicable when multiple layers are involved. If what is specified in subclause 8.4.9 cannot be followed, then the composition time of a sample shall be set as if the sample were an output sample (i.e., at least one picture of the sample is output). If the decoding of a sample results into no output pictures, the presentation of the sample is omitted and the duration of the previous sample is extended, so that the following samples have correct composition timing.

9.6   L-HEVC specific structures

9.6.1   External base layer sample group

9.6.1.1   Definition

When a multi-layer HEVC bitstream uses an external base layer (i.e., when an active VPS of the bitstream has vps_base_layer_internal_flag equal to 0 and vps_base_layer_available_flag equal to 1), an `LhvcExternalBaseLayerInfo` sample grouping may be used to signal parameters that are required for inter-layer prediction from the external base layer to a track containing L-HEVC samples. When a sample is linked to the zero index of this sample grouping, it means that no decoded base layer picture is used for the decoding of that sample.

9.6.1.2   Syntax

```
aligned(8) class LhvcExternalBaseLayerInfo() extends VisualSampleGroupEntry('lbli')
{
   bit(1) reserved = '1'b;
   unsigned int(1) bl_irap_pic_flag;
   unsigned int(6) bl_irap_nal_unit_type;
   signed   int(8) sample_offset;
}
```

9.6.1.3   Semantics

In this subclause, the term current sample refers to the sample that is associated with an entry in the `LhvcExternalBaseLayerInfo` sample group in an L-HEVC track.

`bl_irap_pic_flag` specifies the value of the `BlIrapPicFlag` variable for the associated decoded picture, when that decoded picture is provided as a decoded base layer picture for the decoding of the current sample.

`bl_irap_nal_unit_type` specifies, when `bl_irap_pic_flag` is equal to 1, the value of the nal_unit_type syntax element for the associated decoded picture, when that decoded picture is provided as a decoded base layer picture for the decoding of the current sample.

`sample_offset` gives the relative index of the associated sample in the track containing the base layer (i.e., the track referred to by the `'sbas'` track reference). The decoded picture resulting from the decoding of the associated sample in the track containing the base layer is the associated decoded picture that should be provided for the decoding of the current sample. `sample_offset` equal to 0 specifies that the associated sample has the same, or the closest preceding, decoding time compared to the decoding time of the current sample; `sample_offset` equal to 1 specifies that the associated sample is the next sample relative to the associated sample derived for `sample_offset` equal to 0; `sample_offset` equal to -1 specifies that the associated sample is the previous sample relative to the associated sample derived for sample_offset equal to 0, and so on.

9.6.2   The operating points information sample group

9.6.2.1   Definition

Applications are informed about the different operating points provided by a given L-HEVC bitstream and their constitution by using the Operating Points Information sample group (`'oinf'`). Each operating point is related to an output layer set, a max TemporalId value, and a profile, level and tier signalling. All this information is captured by the `'oinf'` sample group. Apart from this information, this sample group also provides the dependency information between layers, the types of scalabilities coded in

the L-HEVC bit stream, and the dimension identifiers that relate to any particular layer for a given scalability type.

For all tracks of an HEVC bitstream containing only NAL units with nuh_layer_id 0 and for which temporal sublayers are split across several tracks, there is no requirement for any track to carry `'oinf'` sample group.

For all tracks of an L-HEVC bitstream, there shall be exactly one track among this set that carries an `'oinf'` sample group. Except the tracks that carry the `'oinf'` sample group, all other tracks of that L-HEVC bitstream shall have a track reference of type `'oref'` to the track that carries the 'oinf' sample group they relate to.

NOTE   An L-HEVC bitstream usually requires using only one track with `'oinf'` sample group information. However, extractors enable creating new bitstreams with different profiles and levels not signalled in the `'oinf'` of the original bitstream for which data is extracted. Using extractors can require additional `'oinf'` to store the operation information of these new bitstreams.

For any particular sample in a particular track, the temporally collocated sample in another track is defined as the one with the same decoding time as that of this particular sample. For each sample SN in a track TN that has an `'oref'` track reference to the track Tk that carries the `'oinf'` sample group, the follow applies:

— If there is a temporally collocated sample Sk in the track Tk, then the sample SN is associated with the same `'oinf'` sample group entry as the sample Sk.

— Otherwise, the sample SN is associated with the same `'oinf'` sample group entry as the last of the samples in the track Tk that precede the sample SN in decoding time.

When several VPSs are referenced by an L-HEVC bitstream, it may be needed to include several entries in the sample group description box with grouping type `'oinf'`. For more common cases where a single VPS is present, it is recommended to use the default sample group mechanism defined in ISO/IEC 14496-12 and include the Operating Points Information sample group in the sample table box, rather than including it in each track fragment.

The `grouping_type_parameter` is not defined for the `SampleToGroupBox` with grouping type `'oinf'`.

An operating point shall contain at least one other layer in addition to an external base layer.

9.6.2.2   Syntax

```
class OperatingPointsRecord {

   unsigned int(16) scalability_mask;

   bit(2) reserved = 0;

   unsigned int(6) num_profile_tier_level;

   for (i=1; i<=num_profile_tier_level; i++) {

      unsigned int(2) general_profile_space;

      unsigned int(1) general_tier_flag;

      unsigned int(5) general_profile_idc;

      unsigned int(32) general_profile_compatibility_flags;

      unsigned int(48) general_constraint_indicator_flags;

      unsigned int(8) general_level_idc;

   }
```

```
unsigned int(16) num_operating_points;
for (i=0; i<num_operating_points) {
    unsigned int(16) output_layer_set_idx;
    unsigned int(8) max_temporal_id;
    unsigned int(8) layer_count;
    for (j=0; j<layer_count; j++) {
        unsigned int(8) ptl_idx;
        unsigned int(6) layer_id;
        unsigned int(1) is_outputlayer;
        unsigned int(1) is_alternate_outputlayer;
    }
    unsigned int(16) minPicWidth;
    unsigned int(16) minPicHeight;
    unsigned int(16) maxPicWidth;
    unsigned int(16) maxPicHeight;
    unsigned int(2) maxChromaFormat;
    unsigned int(3) maxBitDepthMinus8;
    bit(1) reserved = 0;
    unsigned int(1) frame_rate_info_flag
    unsigned int(1) bit_rate_info_flag
    if (frame_rate_info_flag) {
        unsigned int(16) avgFrameRate;
        bit(6) reserved = 0;
        unsigned int(2) constantFrameRate;
    }
    if (bit_rate_info_flag) {
        unsigned int(32) maxBitRate;
        unsigned int(32) avgBitRate;
    }
}
unsigned int(8) max_layer_count;
for (i=0; i<max_layer_count; i++) {
    unsigned int(8) layerID;
```

```
      unsigned int(8) num_direct_ref_layers;

      for (j=0; j<num_direct_ref_layers; j++) {

         unsigned int(8) direct_ref_layerID;

      }

      for (j=0; j<16; j++) {

         if (scalability mask & (1 << j))

            unsigned int(8) dimension_identifier;

      }

   }

}

class OperatingPointsInformation extends VisualSampleGroupEntry ('oinf') {

   OperatingPointsRecord oinf;

}
```

9.6.2.3   Semantics

scalability_mask: This field indicates the scalability types that are represented by the access units resolved from the samples that are associated with this 'oinf' sample group entry. Each bit in the scalability_mask field denotes a scalability dimension as coded in the scalability_mask_flag of the VPS extension syntax as defined in ISO/IEC 23008-2. The most significant bit in the scalability_mask field corresponds to scalability_mask_flag[0] of the VPS extension syntax. A one in a bit position indicates that the scalability dimension is present.

num_profile_tier_level: Gives the number of following profile, tier, and level combinations as well as the associated fields.

general_profile_space, general_tier_flag, general_profile_idc, general_profile_ compatibility_flags, general_constraint_indicator_flags, and general_level_idc are defined in ISO/IEC 23008-2.

num_operating_points: Gives the number of operating points for which the information follows.

output_layer_set_idx is the index of the output layer set that defines the operating point. The mapping between output_layer_set_idx and the layer_id values shall be the same as specified by the VPS for an output layer set with index output_layer_set_idx.

max_temporal_id: Gives the maximum TemporalId of NAL units of this operating point.

layer_count: This field indicates the number of necessary layers, as defined in ISO/IEC 23008-2, of this operating point.

ptl_idx: Indicates the index, where value 1 is the first entry in the list of profile, tier, level in the OperatingPointsRecord, of the listed profile, level, and tier flags for a layer with identifier equal to layer_id. When the value of ptl_idx equals zero for a layer, that layer shall be assumed to have no profile, level, and tier signalled. ptl_idx shall be equal to 0 for an external base layer. When ptl_idx is equal to 0 for a layer that is not an external base layer, that layer shall not be an output layer or a layer that is a direct or indirect reference layer of any output layer of the operating point.

layer_id: provides the nuh_layer_id values for the layers of the operating point.

is_outputlayer: A flag that indicates if the layer is an output layer or not. A one indicates an output layer.

is_alternate_outputlayer: This flag when set indicates that this layer can be considered as an alternate output layer for this operating point. This flag is set to one if and only if one layer in the operating point has its is_outputlayer flag set.

minPicWidth specifies the least value of the luma width indicators as defined by the pic_width_in_luma_samples parameter in ISO/IEC 23008-2 for the stream of the operating point.

minPicHeight specifies the least value of the luma height indicators as defined by the pic_height_in_luma_samples parameter in ISO/IEC 23008-2 for the stream of the operating point.

maxPicWidth specifies the greatest value of the luma width indicators as defined by the pic_width_in_luma_samples parameter in ISO/IEC 23008-2 for the stream of the operating point.

maxPicHeight specifies the greatest value of the luma height indicators as defined by the pic_height_in_luma_samples parameter in ISO/IEC 23008-2 for the stream of the operating point.

maxChromaFormat specifies the greatest value of the chroma_format indicator as defined by the chroma_format_idc parameter in ISO/IEC 23008-2 for the stream of the operating point.

maxBitDepthMinus8 specifies the greatest value of the luma and chroma bit depth indicators as defined by the bit_depth_vps_luma_minus8 and bit_depth_vps_chroma_minus8 parameters, respectively, in ISO/IEC 23008-2 for the stream of the operating point.

frame_rate_info_flag equal to 0 indicates that no frame rate information is present for the operating point. The value 1 indicates that frame rate information is present for the operating point.

bit_rate_info_flag equal to 0 indicates that no bitrate information is present for the operating point. The value 1 indicates that bitrate information is present for the operating point.

avgFrameRate gives the average frame rate in units of frames/(256 seconds) for the operating point. Value 0 indicates an unspecified average frame rate.

constantFrameRate equal to 1 indicates that the stream of the operating point is of constant frame rate. Value 2 indicates that the representation of each temporal layer in the stream of the operating point is of constant frame rate. Value 0 indicates that the stream of the operating point may or may not be of constant frame rate.

maxBitRate gives the maximum bit rate in bits/second of the stream of the operating point, over any window of one second.

avgBitRate gives the average bit rate in bits/second of the stream of the operating point.

max_layer_count: The count of all unique non-base layers in all of the operating points.

layerID: nuh_layer_id of a layer for which the all the direct reference layers are given in the following loop of direct_ref_layerID.

num_direct_ref_layers: The number of direct reference layers for the layer with nuh_layer_id equal to layerID.

direct_ref_layerID: nuh_layer_id of the direct reference layer.

dimension_identifier: Set to the value of the dimension_id field as specified in the VPS extension syntax as defined in ISO/IEC 23008-2.

9.6.3 The layer information sample group

9.6.3.1 Definition

The list of layers and sub layers a track carries is signalled in the Layer Information sample group. Every L-HEVC track, including the base track (when coded with HEVC), shall carry a 'linf' sample group.

When several VPSs are referenced by an L-HEVC bitstream, it may be needed to include several entries in the sample group description box with grouping_type 'linf'. For more common cases where a single VPS is present, it is recommended to use the default sample group mechanism defined in ISO/IEC 14496-12 and include the Layer Information sample group in the sample table box, rather than including it in each track fragment.

The grouping_type_parameter is not defined for the SampleToGroupBox with grouping type 'linf'.

9.6.3.2 Syntax

```
class LayerInfoGroupEntry extends VisualSampleGroupEntry ('linf')) {

    bit(2) reserved = 0;

    unsigned int (6) num_layers_in_track;

    for (i=0; i<num_layers_in_track; i++) {

        bit(4) reserved = 0;

        unsigned int (6) layer_id;

        unsigned int (3) min_TemporalId;

        unsigned int (3) max_TemporalId;

        bit(1) reserved = 0;

        unsigned int (7) sub_layer_presence_flags;

    }

}
```

9.6.3.3 Semantics

num_layers_in_track: The number of layers carried in any sample of this track associated with this sample group.

layer_id: nuh_layer_id of the layer carried in the associated samples. The instances of this field shall be in ascending order of layer_id in the loop.

min_TemporalId: The minimum TemporalId value for the sub-layers in the layer within the track.

max_TemporalId: The maximum TemporalId value for the sub-layers in the layer within the track.

sub_layer_presence_flags: Each bit of this field at bit position bitPos in the range of min_TemporalId to max_TemporalId indicates that the sub-layer with TemporalId equal to bitPos is natively present (when the bit is equal to 1) or present by extractors (when the bit is equal to 0) in the track. Bits of this field at bit positions less than min_TemporalId or greater than max_TemporalId are unspecified.

9.6.4 The layer information sample group

9.6.4.1 Definition

Each decoding time hint sample group description entry ('opth') records a delta time in terms of the clock ticks (given by timescale of the MediaHeaderBox). The corrected decoding time is defined as the sum of the delta time associated with a sample through the SampleToGroupBox of type 'opth' and the decoding time of the sample. The corrected decoding times conform to the hypothetical reference decoder of ISO/IEC 23008-2 operating according to a partitioning scheme where each layer is in its own bitstream partition, as defined in ISO/IEC 23008-2.

All `SampleToGroupBoxes` for the decoding time hint sample group shall include `grouping_type_parameter`. The `grouping_type_parameter` field is specified for the decoding time hint sample group as follows:

```
unsigned int(16) reserved = 0;

unsigned int(16) operating_point_index;
```

`operating_point_index` specifies the index of the operating point, as given in the associated `OperatingPointsInformation` sample group description, for which this sample group provides the corrected decoding times. A value of 0 indicates the first operating point in that sample group description.

9.6.4.2 Syntax

```
class OperatingPointDecodeTimeHint()

extends VisualSampleGroupEntry ('opth')

{

    signed int(32) delta_time;

}
```

9.6.4.3 Semantics

`delta_time` plus the decoding time (derived from the `TimeToSampleBox` and `TrackRunBoxes`, if any) provides the corrected decoding time of the associated sample. Time-scale units, as given by `timescale` of the `MediaHeaderBox` of this track, are used for the calculation of `delta_time`.

10 Storage of tiled HEVC and L-HEVC video streams

10.1 Overview

ISO/IEC 23008-2 video provides support for coding of rectangular regions called tiles. HEVC tiles do not have coding dependencies with other HEVC tiles in the same coded picture but may have coding dependencies with other HEVC tiles from previous coded pictures or may be independently decoded. This section defines tools to describe and manipulate tiles, including:

— Description of HEVC tiles and their temporal coding dependencies with other HEVC tiles,

— Track(s) extracting one or several tiles from other track(s), and

— Track(s) containing only data from one or several HEVC tiles for fast access to a tile region over a network.

A tile region is defined as one or more complete HEVC tiles in one or more complete slice segments that are within the same coded picture and that contain no other HEVC tiles. The slice segments of a tile region may but need not be contiguous in decoding order. A tile region covers a rectangle without holes. Tile regions within a picture do not overlap with each other.

Tile regions can be described through tile region visual sample group description entries (i.e., instances of `TileRegionGroupEntry`) with `tile_region_flag` equal to 1.

If each sample of a track consists of NAL units of only one tile region, `SampleToGroupBox` of type `'trif'` can be used to associate samples to the tile region, but this `SampleToGroupBox` of type `'trif'` can be omitted if the default sample grouping mechanism is used (i.e., when the version of the `SampleGroupDescriptionBox` of type `'trif'` is equal to or greater than 2). Otherwise, samples, NAL units, and tile regions are associated with each other through `SampleToGroupBox`es of type `'nalm'`

and `grouping_type_parameter` equal to `'trif'` and `SampleGroupDescriptionBox` of type `'nalm'`.A `TileRegionGroupEntry` describes:

— a tile region, and

— coding dependencies between this tile region and other tile regions.

Each `TileRegionGroupEntry` is assigned a unique identifier, called `groupID`. This identifier can be used to associate NAL units in a sample to a particular `TileRegionGroupEntry`.

Positioning and size of tile regions are identified using luma sample coordinates.

When used with movie fragments, `TileRegionGroupEntry` can be defined for the duration of the movie fragment, by defining a new `SampleGroupDescriptionBox` in the track fragment box as defined in ISO/IEC 14496-12:2015, 8.9.4. However, there shall not be any `TileRegionGroupEntry` in a track fragment that has the same `groupID` as a `TileRegionGroupEntry` already defined.

Sub-tracks can be defined by grouping the different tile regions for this sub-track, using a `TileSubTrackGroupBox` in the sub-track definition box.

NAL units mapped to a tile region may either be carried in the video track, as usual, or in a separate track called tile track. Tile tracks are defined for both HEVC and L-HEVC.

10.2 NAL unit map entry

10.2.1 Definition

The `NALUMapEntry` may be used to assign an identifier, called `groupID`, to each NAL unit. The `NALUMapEntry`, when present, shall be linked to a sample group description providing the semantics of that `groupID`. This link shall be provided by setting the `grouping_type_parameter` of the `SampleToGroupBox` of type `'nalm'` to the four-character code of the associated sample grouping type. Consequently, a `SampleToGroupBox` of type `'nalm'` shall never use version 0 of the box. It is forbidden to indicate an associated `grouping_type` for which the sample group description definition does not provide semantics for `groupID`.

In case of HEVC tiling, the `SampleToGroupBox` of type `'nalm'` describing the tiling of samples shall have its `grouping_type_parameter` set to `'trif'`.

A track shall not contain both a `SampleToGroupBox` of type `'nalm'` associated with `grouping_type_parameter` equal to a particular value groupType and `SampleToGroupBox` of type groupType. When a track contains a `SampleToGroupBox` of type `'nalm'` associated with `grouping_type_parameter` groupType, NAL units of the mapped sample are indirectly associated with the sample group description of type groupType through the `groupID` of the `NALUMapEntry` applicable for that sample. When a track contains a `SampleToGroupBox` of type groupType, each sample is directly mapped to the sample group description of type groupType through the `SampleToGroupBox` of type groupType and all NAL units of the mapped sample are associated with the same `groupID`.

10.2.2 Syntax

```
class NALUMapEntry() extends VisualSampleGroupEntry ('nalm') {

   bit(6) reserved = 0;

   unsigned int(1) large_size;

   unsigned int(1) rle;

   if (large_size) {

      unsigned int(16) entry_count;

   } else {
```

```
      unsigned int(8) entry_count;

   }

   for (i=1; i<= entry_count; i++) {

      if (rle) {

         if (large_size) {

            unsigned int(16) NALU_start_number;

         } else {

            unsigned int(8) NALU_start_number;

         }

      }

      unsigned int(16) groupID;

   }

}
```

10.2.3 Semantics

large_size indicates whether the number of NAL units entries in the track samples is represented on 8 or 16 bits.

rle indicates whether run-length encoding is used (1) to assign groupID to NAL units or not (0).

entry_count specifies the number of entries in the map. Note that when rle is equal to 1, the entry_count corresponds to the number of runs where consecutives NAL units are associated with the same group. When rle is equal to 0, entry_count represents the total number of NAL units.

NALU_start_number is the 1-based NAL unit index in the sample of the first NAL unit in the current run associated with groupID.

groupID specifies the unique identifier of the group. More information about the group is provided by the sample group description entry with this groupID and grouping_type equal to the grouping_type_parameter of the SampleToGroupBox of type 'nalm'.

All NAL units aggregated by an aggregator by inclusion or by reference count as a single NAL unit in the indexing provided by the NALUMapEntry.

When used with movie fragments, new NAL unit maps can be defined for the duration of the movie fragments, by defining a new SampleGroupDescription box in the track fragment, box as defined in ISO/IEC 14496-12:2015, 8.9.4.

10.3 Tile region group entry

10.3.1 Definition

Group Type: 'trif'

Container: SampleGroupDescriptionBox ('sgpd')

Mandatory: No

Quantity: Zero or more

The `TileRegionGroupEntry` may be used to describe a tile region. The `grouping_type_parameter` is not defined for the `SampleToGroupBox` with grouping type `'trif'`.

10.3.2 Syntax

```
class TileRegionGroupEntry() extends VisualSampleGroupEntry ('trif')
{
   unsigned int(16) groupID;
   unsigned int(1) tile_region_flag;
   if (!tile_region_flag)
      bit(7) reserved = 0;
   else {
      unsigned int(2) independent_idc;
      unsigned int(1) full_picture;
      unsigned int(1) filtering_disabled;
      unsigned int(1) has_dependency_list;
      bit(2) reserved = 0;
      if (!full_picture) {
         unsigned int(16) horizontal_offset;
         unsigned int(16) vertical_offset;
   }
   unsigned int(16) region_width;
   unsigned int(16) region_height;
   if (has_dependency_list) {
      unsigned int(16) dependency_tile_count;
      for (i=1; i<= dependency_tile_count; i++)
         unsigned int(16) dependencyTileGroupID;
   }
}
```

10.3.3 Semantics

`groupID` is a unique identifier for the tile region group described by this sample group entry. The value of `groupID` in a tile region group entry shall be greater than 0. The value 0 is reserved for a special use.

When there is `SampleToGroupBox` of type `'nalm'` and `grouping_type_parameter` equal to `'trif'`, a `SampleGroupDescriptionBox` of type `'trif'` shall be present, and the following applies:

— The value of `groupID` in a tile region group entry shall be equal to the `groupID` in one of the entries of `NALUMapEntry`.

— A NAL unit being mapped to `groupID` 0 by a `NALUMapEntry` implies that the NAL unit is required for decoding any tile region in the same coded picture as this NAL unit.

NOTE 1   There can be multiple tile region group entries with the same values of `horizontal_offset`, `vertical_offset`, `region_width` and `region_height`, respectively, but with different `groupID` values, for describing varying dependencies.

`tile_region_flag` equal to 1 specifies that the region covered by the NAL units within a picture and associated with this tile region group entry is a tile region, and further information of the tile region is provided by subsequent fields in this tile region group entry. The value 0 specifies that the region covered by the NAL units within a picture and associated with this tile region group entry is not a tile region, and no further information of the region is provided in this tile region group entry.

When a multi-layer bitstream is carried in one or more tracks, for any two layers layerA and layerB of the bitstream, the following constraint applies: When a NAL unit of layerA is associated with a `groupID` value gIdA for which the corresponding `tile_region_flag` is equal to 1, and a NAL unit of layerB is associated with a `groupID` value gIdB for which the corresponding `tile_region_flag` is equal to 1, gIdA shall not be equal to gIdB.

`independent_idc` specifies the coding dependencies between each tile region associated with this tile region group entry and other tile regions in the same picture or in reference pictures of the same layer. Inter-layer dependencies, if any, are indicated by a list of `dependencyTileGroupID` (when `has_dependency_list` is equal to 1).

This field takes the following values:

— If `independent_idc` equals 0, the coding dependencies between this tile region and other tile regions in the same picture or in reference pictures of the same layer is either described by a list of `dependencyTileGroupID` (when `has_dependency_list` is equal to 1) or unknown (when `has_dependency_list` is equal to 0).

— If `independent_idc` equals 1, there are no temporal dependencies between this tile region and the tile regions with different `groupID` in any reference pictures of the same layer but there can be coding dependencies between this tile region and a tile region with the same `groupID` in a reference picture of the same layer.

— If `independent_idc` equals 2, there are no coding dependencies between this tile region and any tile region in a reference picture of the same layer.

— The value 3 is reserved.

`full_picture`, when set, indicates that each tile region associated with this tile region group entry is a complete picture, in which case `region_width` and `region_height` shall be set to the width and height, respectively, of the complete picture, and `independent_idc` shall be set to 1 or 2.

`filtering_disabled`, when set, indicates that for each tile region associated with this tile region group entry the in-loop filtering operation does not require access to pixels adjacent to this tile region, i.e., bit-exact reconstruction of the tile region is possible without decoding the adjacent tiles.

`has_dependency_list`, when set to 1, indicates that `dependency_tile_count` and, when `dependency_tile_count` is greater than 0, a list of `dependencyTileGroupID` are present. When set to 0, `dependency_tile_count` is not present and no `dependencyTileGroupID` is present.

`horizontal_offset` and `vertical_offset` give respectively the horizontal and vertical offsets of the top-left pixel of the rectangular region that is covered by the tiles in each tile region associated with this tile region group entry, relative to the top-left pixel of the base region, in luma samples. For HEVC and L-HEVC tile tracks as defined in this document, the base region used in the `TileRegionGroupEntry` is the picture to which the tiles in a tile region associated with this tile region group entry belongs.

region_width and region_height give respectively the width and height of the rectangular region that is covered by the tiles in each tile region associated with this tile region group entry, in luma samples.

NOTE 2   For L-HEVC streams using spatial scalability and tiling on both the base and enhancement layers, when each layer is carried in its own track, the TileRegionGroupEntry sample descriptions of the base layer will give coordinates expressed in luma samples of the base layer, while the TileRegionGroupEntry sample descriptions of an enhancement layer will give coordinates expressed in luma samples of the enhancement layer.

dependency_tile_count indicates the number of tile regions each tile region associated with this tile region group entry depends on.

dependencyTileGroupID gives the groupID of a tile region (as defined by a TileRegionGroupEntry) that this tile region depends on. For a particular tile region associated with this tile region group entry, the tile regions it depends on may be from the same layer or the reference layers.

10.4 Tile sub track definition

10.4.1 Overview

A tile sub track describes one or more tile regions.

An additional descriptive attribute 'tile' may be used to indicate the sub track is a spatial part of the track.

Name	Attribute	Description
Tiling	'tile'	The sub-track is spatial part or tile of the track.

A tile sub track is defined as a group of tile regions using the TileSubTrackGroupBox.

10.4.2 TileSubTrackGroupBox

10.4.2.1 Definition

Box Type: 'tstb'

Container: Sub Track Definition box ('strd')

Mandatory: No

Quantity: Zero or one

10.4.2.2 Syntax

```
aligned(8) class TileSubTrackGroupBox extends FullBox('tstb', 0, 0) {

   unsigned int(16) item_count;

   for(i = 0; i< item_count; i++) {

      unsigned int(16) tileGroupID;

   }

}
```

10.4.2.3 Semantics

The union of tileGroupIDs in this box describes the sub track defined by this box.

item_count counts the number of tile groups listed in this box.

tileGroupID is the identifier of the tile region group (groupID) contained in this sub track. groupID is defined in TileRegionGroupEntry.

10.5 HEVC and L-HEVC tile track

10.5.1 Overview

There are cases where storing NAL units of HEVC or L-HEVC tile regions in different tracks are useful for easy access to one or a few particular tile regions. For such cases, tile tracks may be created using the HEVCTileSampleEntry or LHEVCTileSampleEntry sample description format.

An HEVC or L-HEVC tile track is a video track in which there is a 'tbas' track reference to the HEVC or L-HEVC track, respectively. This HEVC or L-HEVC track is refered to as the HEVC or L-HEVC tile base track of the HEVC or L-HEVC tile track, respectively. The sample entry type for an HEVC tile track is 'hvt1'. The sample entry type for an L-HEVC tile track is 'lht1'. The use of sample entry types for tile base tracks is defined in subclause 10.5.5.

A tile track or tile base track shall not include extractors. A tile base track shall not include VCL NAL units. A tile track shall not carry VCL NAL units belonging to more than one layer.

For any picture unit carried by samples in a tile base track and a number of tile tracks, all the NAL units that apply to the entire coded picture shall be carried in the tile base track. These NAL units include (but are not limited to) parameter sets (VPS, SPS, and PPS) as well as EOB and EOS NAL units, when present. The SampleDescriptionBox of a tile track shall not carry NAL units that apply to any entire coded picture, either. The NAL units that do not apply to a tile shall not be carried in the tile track containing that tile. The NAL units that apply to a tile shall be carried in the tile track containing that tile.

NOTE 1   When a NAL unit applies to multiple tiles, these tiles can only be stored in the same tile track.

A tile track shall only depend on the tile base track and shall be independent from any other tile track that includes VCL NAL units of the same layer as this tile track.

An HEVC or L-HEVC sample of a tile track consists of one or more complete tiles in one or more complete slice segments.

An HEVC or L-HEVC sample in the tile base track is considered as a sync sample if and only if the sample resulting from the merging of the corresponding samples of all tile tracks and the tile base track, as defined in subclause 10.5.4, is a sync sample as defined for a non tiled HEVC or L-HEVC track, respectively, that has the sample entry type equal to that in the tile base track and the same configuration records as those in the tile base track.

A sample stored in a tile track is considered as a sync sample when the respective sample in the tile base track is a sync sample.

The SubSampleInformationBox and sample groupings defined for regular HEVC or L-HEVC tracks also apply to HEVC or L-HEVC tile tracks, respectively.

NOTE 2   Many aspects of the tile track can be discovered by inspecting the base track if they are not explicitly declared in the tile track; for example, a 'roll' sample group.

NOTE 3   An implementation can decide to decode only a subset of the tiles of an HEVC or L-HEVC stream. In this case, it can use the dependency information in the TileRegionGroupEntry sample group descriptions to discard un-needed NAL units.

10.5.2 Sample entry name and format for HEVC tile tracks

10.5.2.1 Definition

Sample Entry Type: `'hvt1'`

Container: Sample Description Box (`'stsd'`)

Mandatory: No

Quantity: Zero or more sample entries may be present

This sample entry describes media samples of an HEVC tile track. The `width` and `height` of the `VisualSampleEntry` for an HEVC tile track (sample entry type `'hvt1'`) shall be set to the width and height of the minimum bounding box enclosing all tile regions contained in the track. The layout information in the track header (i.e., `layer`, `matrix`, `width` and `height`) of an HEVC tile track shall be ignored by file parsers. `CleanApertureBox` and `PixelAspectRatioBox` shall not be present in an `'hvt1'` sample description.

10.5.2.2 Syntax

```
class HEVCTileSampleEntry() extends VisualSampleEntry ('hvt1'){

    HEVCTileConfigurationBox   config(); // optional

}

class HEVCTileConfigurationBox extends Box('hvtC') {

    HEVCTileTierLevelConfigurationRecord() HEVCTileTierLevelConfig;

}

aligned(8) class HEVCTileTierLevelConfigurationRecord {

    unsigned int(8) configurationVersion = 1;

    bit(7) reserved = 0;

    unsigned int(1) mcts_tier_flag;

    unsigned int(8) mcts_level_idc;

    bit(8) reserved = 0;

}
```

10.5.2.3 Semantics

The `HEVCTileSampleEntry` shall not contain any `HEVCConfigurationBox`, `LHEVCConfigurationBox` or `MPEG4ExtensionDescriptorsBox`; these boxes are found in the tile base track's sample description. Other optional boxes may be included. There are usually as many entries in the `SampleDescriptionBox` of an HEVC tile track as there are entries in the `SampleDescriptionBox` of the tile base track.

Optionally, the `HEVCTileSampleEntry` may contain one `HEVCTileTierLevelConfigurationRecord`, used to indicate the tier and level information in the case the tile region in this tile track is a motion-constrained tile or tile set.

`Compressorname` in the base class `VisualSampleEntry` indicates the name of the compressor used with the value "`\020HEVC Tile Coding`" being recommended; the first byte is a count of the remaining bytes, here represented by \020, which (being octal 20) is 16 (decimal), the number of bytes in the rest of the string.

mcts_tier_flag, mcts_level_idc are set to the values of the fields mcts_tier_flag and mcts_level_idc in the temporal motion-constrained tile sets SEI message if the tile region in the tile track is a temporal motion-constrained tile set as defined in ISO/IEC 23008-2. If the tile region in the tile track is not the same as any of the temporal motion-constrained tile set in the stream or no temporal motion-constrained tile sets SEI message is present in the stream, an appropriate value according to the temporal motion-constrained tile sets SEI as defined in ISO/IEC 23008-2 shall be set.

10.5.3 Sample entry name and format for L-HEVC tile tracks

10.5.3.1 Definition

Sample Entry Type: 'lht1'

Container: Sample Description Box ('stsd')

Mandatory: No

Quantity: Zero or more sample entries may be present

This sample entry describes media samples of an L-HEVC tile track. The width and height of the VisualSampleEntry for an L-HEVC tile track (sample entry type 'lht1') shall be set to the width and height of the minimum bounding box enclosing all tile regions of all layers contained in the track. The layout information in the track header (i.e., layer, matrix, width and height) of an L-HEVC tile track shall be ignored by file parsers. CleanApertureBox and PixelAspectRatioBox shall not be present in an 'lht1' sample description.

10.5.3.2 Syntax

```
class LHEVCTileSampleEntry() extends VisualSampleEntry ('lht1'){

}
```

10.5.3.3 Semantics

The LHEVCTileSampleEntry shall not contain any LHEVCConfigurationBox, HEVCConfigurationBox or MPEG4ExtensionDescriptorsBox; these boxes are found in the tile base track's sample description. Other optional boxes may be included. There are usually as many entries in the SampleDescriptionBox of an L-HEVC tile track as there are entries in the SampleDescriptionBox of the tile base track.

Compressorname in the base class VisualSampleEntry indicates the name of the compressor used with the value "\022L-HEVC Tile Coding" being recommended; the first byte is a count of the remaining bytes, here represented by \022, which (being octal 22) is 18 (decimal), the number of bytes in the rest of the string.

10.5.4 Bitstream reconstruction from tile base and tile tracks

Since extractors cannot be used in a tile base track, a tile base track shall indicate the tile ordering using a 'sabt' track reference to the tile tracks. The 'sabt' track reference may only be used to refer to tile tracks from a tile base track. When the 'sabt' track reference is present in a track, the samples of this track shall not use aggregators.

The bitstream is reconstructed as follows:

1. For a tile base track and a number of tile tracks carrying VCL NAL units of one layer, a picture unit is firstly reconstructed to consist of the following NAL units in the order listed:

   a) If the picture unit corresponds to a sample that is the first sample of a set of samples associated with a sample entry, the parameter sets and SEI NAL units contained in the sample entry

b) NAL units in the sample of the tile base track

c) NAL units in the samples of the tile tracks in the order of the `'sabt'` track references

Then the following steps apply, in the order listed, for the above-reconstructed picture unit:

d) If there is one or more than one EOS NAL unit present, an EOS NAL unit is placed at the end of the picture unit and any other EOS NAL unit is removed.

e) If there is one or more than one EOB NAL unit present, one EOB NAL unit is placed at the end of the picture unit; any other EOB NAL unit is removed.

2. If only one layer is involved, the reconstructed picture unit is the access unit. Otherwise, the access unit is reconstructed from all the picture units of the involved layers as specified in subclause 9.5.2.2.

3. Finally, the bitstream is reconstructed from the reconstructed access units as specified in subclause 9.5.2.2.

NOTE   Picture units, access units, and bitstreams reconstructed as above are not always conforming to ISO/IEC 23008-2. For example, for a picture unit reconstructed as above, some VCL NAL units covering some tiles of the picture may be missing due to that the regions represented by those tiles are not need by the application.

10.5.5 Sample entry names for tile base tracks

When no NAL unit is included in a sample in a tile base track (i.e. all the NAL units are in the tile tracks), the size of the sample may be 0, and the sample documents the timing and other sample properties of the sample.

The sample entry names of a tile base track and the corresponding tile tracks shall be one of the rows in Table 13.

Table 13 — Sample entry names for a tile base track and the corresponding tile tracks

Sample entry name for a tile base track	Sample entry name for a corresponding tile track
hvc2	hvt1
hev2	hvt1
lhv1	lht1
lhe1	lht1

Annex A
(normative)

In-stream structures

A.1 General

Aggregators and Extractors are file format internal structures enabling efficient grouping of NAL units or extraction of NAL units from other tracks.

Aggregators and Extractors use a syntax that is similar to the NAL unit syntax but does not follow the start code emulation prevention mechanism required for the NAL unit syntax as specified in ISO/IEC 14496-10 or ISO/IEC 23008-2. These NAL-unit-like structures are seen as NAL units in the context of the sample structure. While accessing a sample, Aggregators shall be removed (leaving their contained or referenced NAL units) and Extractors shall be replaced by the data they reference. Aggregators and Extractors shall not be output by file parsers.

These structures use NAL unit types reserved for the application/transport layer by ISO/IEC 14496-10 or ISO/IEC 23008-2.

See Annex F for more information about use of "reserved", "unspecified", "not specified" and "registrant-defined" `nal_unit_type` values.

A.2 Aggregators

A.2.1 Definition

This subclause describes Aggregators, which enable NALU-map-group entries to be consistent and repetitive (in accordance with Annex B).

Aggregators are used to group NAL units belonging to the same sample.

For storage of ISO/IEC 14496-10 video, the following rules apply:

— Aggregators use the same NAL unit header as SVC VCL NAL units, MVC VCL NAL units, MVC+D depth VCL NAL units, or 3D-AVC VCL NAL units, but with a different value of NAL unit type.

— If the sample entry contains the MVCD Configuration Box or the A3D Configuration Box, the NAL unit header of the aggregator follows the syntax of the NAL unit header for the NAL unit of nal_unit_type equal to 21. Otherwise, the NAL unit header of the aggregator follows the syntax of the NAL unit header for the NAL unit of nal_unit_type equal to 20.

— If the sample entry contains the SVC Configuration Box and the svc_extension_flag of the NAL unit syntax (specified in ISO/IEC 14496-10:2014, 7.3.1) of an aggregator is equal to 1, the NAL unit header of SVC VCL NAL units is used for the aggregator.

— Otherwise, if the sample entry contains the MVCD Configuration Box or the A3D Configuration Box and additionally the avc_3d_extension_flag of the NAL unit syntax of an aggregator is equal to 1, the NAL unit header of 3D-AVC VCL NAL units is used for the aggregator.

— Otherwise, the NAL unit header of MVC and MVC+D depth VCL NAL units is used for the aggregator.

For storage of ISO/IEC 23008-2 video, Aggregators use the NAL unit header as defined in ISO/IEC 23008-2, which has the same syntax for plain HEVC and layered HEVC. An ISO/IEC 23008-2

aggregator shall not aggregate NAL units belonging to more than one picture unit (where picture unit is defined in ISO/IEC 23008-2).

Aggregators can both aggregate, by *inclusion,* NAL units within them (within the size indicated by their length) and also aggregate, by *reference,* NAL units that follow them (within the area indicated by the additional_bytes field within them). When the stream is scanned by a file reader that does not support aggregators, only the included NAL units are seen as "within" the aggregator. This permits such a file reader to skip a whole set of un-needed NAL units when they are aggregated by inclusion. This also permits such a file reader not to skip NAL units but let them remain in-stream when they are aggregated by reference.

Aggregators can be used to group base layer or base view NAL units. If these Aggregators are used in an 'avc1', 'hvc1', or 'hev1' track then an aggregator shall not use inclusion but reference of base layer or base view NAL units (the length of the Aggregator includes only its header and the NAL units referenced by the Aggregator are specified by additional_bytes).

When the aggregator is referenced by either an extractor with data_length equal to zero, or by a Map sample group, the aggregator is treated as aggregating both the included and referenced bytes.

An Aggregator may include or reference Extractors. An Extractor may extract from Aggregators. An aggregator shall not include or reference another aggregator directly; however, an aggregator may include or reference an extractor that references an aggregator.

An Aggregator shall not be empty, i.e., it shall include or reference to at least one NAL unit.

When scanning the stream:

a)   if the aggregator is unrecognized (e.g. by an AVC or HEVC reader or decoder) it is easily discarded with its included content;

b)   if the aggregator is not needed (i.e. it belongs to an undesired layer) it and its contents both by inclusion and reference are easily discarded (using its length and additional_bytes fields);

c)   if the aggregator is needed, its header is easily discarded and its contents retained.

An aggregator is stored within a sample like any other NAL unit.

All NAL units remain in decoding order within an aggregator.

A.2.2   Syntax

```
class aligned(8) Aggregator (AggregatorSize) {

   NALUnitHeader();

   unsigned int i = sizeof(NALUnitHeader());

   unsigned int((lengthSizeMinusOne+1)*8)

      additional_bytes;

   i += lengthSizeMinusOne+1;

   while (i<AggregatorSize) {

      unsigned int((lengthSizeMinusOne+1)*8)

         NALUnitLength;

      unsigned int(NALUnitLength*8) NALUnit;

      i += NALUnitLength+lengthSizeMinusOne+1;
```

```
        }

}
```

NOTE   The syntax of Aggregators does not always follow the NAL unit syntax and the NAL unit constraints specified in ISO/IEC 14496-10 or ISO/IEC 23008-2. For example, there can be three continuous bytes equal to a value in the range of 0x000000 to 0x000010, inclusive. This specifiation disallows the presence of Aggregators in a video bitstream output from parsing a file, therefore formal non-compliance with the video specifications is immaterial as they will never be presented to a video decoder.

A.2.3 Semantics

The value of the variable AggregatorSize is equal to the size of the aggregator, and the function sizeof(X) returns the size of the field X in bytes.

`NALUnitHeader()`: the first four bytes of SVC, MVC, and MVC+D depth VCL NAL units, or the first three bytes of 3D-AVC NAL units, or the first two bytes of ISO/IEC 23008-2 NAL units.

`nal_unit_type` shall be set to 30 for ISO/IEC 14496-10 video and 48 for ISO/IEC 23008-2 video.

For an aggregator including or referencing SVC NAL units, the following shall apply.

`forbidden_zero_bit` and `reserved_three_2bits` shall be set as specified in ISO/IEC 14496-10.

Other fields (`nal_ref_idc`, `idr_flag`, `priority_id`, `no_inter_layer_pred_flag`, `dependency_id`, `quality_id`, `temporal_id`, `use_ref_base_pic_flag`, `discardable_flag`, and `output_flag`) shall be set as specified in A.4.

For an aggregator including or referencing MVC or MVC+D depth NAL units, the following shall apply.

`forbidden_zero_bit` and `reserved_one_bit` shall be set as specified in ISO/IEC 14496-10.

Other fields (`nal_ref_idc`, `non_idr_flag`, `priority_id`, `view_id`, `temporal_id`, `anchor_pic_flag`, and `inter_view_flag`) shall be set as specified in A.5.

For an aggregator including or referencing 3D-AVC NAL units, the following shall apply.

`forbidden_zero_bit` and `reserved_three_2bits` shall be set as specified in ISO/IEC 14496-10.

Other fields (`nal_ref_idc`, `view_idx`, `depth_flag`, `non_idr_flag`, `temporal_id`, `anchor_pic_flag`, and `inter_view_flag`) shall be set as specified in A.6.

For an aggregator including or referencing ISO/IEC 23008-2 NAL units, the following shall apply.

`forbidden_zero_bit` shall be set as specified in ISO/IEC 23008-2.

Other fields (`nuh_layer_id` and `nuh_temporal_id_plus1`) shall be set as specified in A.8.

`additional_bytes`: The number of bytes following this aggregator that should be considered as aggregated when this aggregator is referenced by an extractor with data_length equal to zero or Map sample group.

`NALUnitLength`: Specifies the size, in bytes, of the NAL unit following. The size of this field is specified with the `lengthSizeMinusOne` field.

`NALUnit`: a NAL unit as specified in ISO/IEC 14496-10 or ISO/IEC 23008-2, including the NAL unit header. The size of the NAL unit is specified by `NALUnitLength`.

A.3 Extractors for SVC, MVC, and MVD tracks

A.3.1 Definition

This subclause describes Extractors for SVC, MVC, or MVD tracks. Extractors enable compact formation of tracks that extract, by reference, NAL unit data from other tracks.

An Aggregator may include or reference Extractors. An Extractor may reference Aggregators. When an extractor is processed by a file reader that requires it, the extractor is logically replaced by the bytes it references. Those bytes shall not contain extractors; an extractor shall not reference, directly or indirectly, another extractor.

NOTE   The track that is referenced can contain extractors even though the data that is referenced by the extractor shall not.

An extractor contains an instruction to extract data from another track, which is linked to the track in which the extractor resides, by means of a track reference of type 'scal'.

The bytes copied shall be one of the following:

a)   One entire NAL unit; note that when an Aggregator is referenced, both the included and referenced bytes are copied

b)   More than one entire NAL unit

In both cases the bytes extracted start with a valid length field and a NAL unit header.

The bytes are copied only from the single identified sample in the track referenced through the indicated 'scal' track reference. The alignment is on decoding time, i.e. using the time-to-sample table only, followed by a counted offset in sample number. Extractors are a media-level concept and hence apply to the destination track before any edit list is considered. (However, one would normally expect that the edit lists in the two tracks would be identical).

A.3.2 Syntax

```
class aligned(8) Extractor () {

   NALUnitHeader();

   unsigned int(8) track_ref_index;

   signed int(8) sample_offset;

   unsigned int((lengthSizeMinusOne+1)*8)

      data_offset;

   unsigned int((lengthSizeMinusOne+1)*8)

      data_length;

}
```

NOTE   The syntax of Extractors does not always follow the NAL unit syntax and the NAL unit constraints specified in ISO/IEC 14496-10. For example, there can be three continuous bytes equal to a value in the range of 0x000000 to 0x000010, inclusive. This specifiation disallows the presence of Extractors in a video bitstream output from parsing a file, therefore formal non-compliance with the video specifications is immaterial as they will never be presented to a video decoder.

A.3.3 Semantics

NALUnitHeader(): the first four bytes of SVC, MVC and MVC+D depth VCL NAL units, or the first three bytes of 3D-AVC NAL units.

`nal_unit_type` shall be set to 31 for ISO/IEC 14496-10 video.

For an extractor referencing SVC NAL units, the following shall apply.

`forbidden_zero_bit` and `reserved_three_2bits` shall be set as specified in ISO/IEC 14496-10.

Other fields (`nal_ref_idc`, `idr_flag`, `priority_id`, `no_inter_layer_pred_flag`, `dependency_id`, `quality_id`, `temporal_id`, `use_ref_base_pic_flag`, `discardable_flag`, and `output_flag`) shall be set as specified in A.4.

For an extractor referencing MVC or MVC+D depth NAL units, the following shall apply.

`forbidden_zero_bit` and `reserved_one_bit` shall be set as specified in ISO/IEC 14496-10.

Other fields (`nal_ref_idc`, `non_idr_flag`, `priority_id`, `view_id`, `temporal_id`, `anchor_pic_flag`, and `inter_view_flag`) shall be set as specified in A.5.

For an extractor referencing 3D-AVC NAL units, the following shall apply.

`forbidden_zero_bit` and `reserved_three_2bits` shall be set as specified in ISO/IEC 14496-10.

Other fields (`nal_ref_idc`, `view_idx`, `depth_flag`, `non_idr_flag`, `temporal_id`, `anchor_pic_flag`, and `inter_view_flag`) shall be set as specified in A.6.

`track_ref_index` specifies the index of the track reference of type `'scal'` to use to find the track from which to extract data. The sample in that track from which data is extracted is temporally aligned or nearest preceding in the media decoding timeline, i.e. using the time-to-sample table only, adjusted by an offset specified by sample_offset with the sample containing the Extractor. The first track reference has the index value 1; the value 0 is reserved.

`sample_offset` gives the relative index of the sample in the linked track that shall be used as the source of information. Sample 0 (zero) is the sample with the same, or the closest preceding, decoding time compared to the decoding time of the sample containing the extractor; sample 1 (one) is the next sample, sample -1 (minus 1) is the previous sample, and so on.

`data_offset`: The offset of the first byte within the reference sample to copy. If the extraction starts with the first byte of data in that sample, the offset takes the value 0. The offset shall reference the beginning of a NAL unit length field.

`data_length`: The number of bytes to copy. If this field takes the value 0, then the entire single referenced NAL unit is copied (i.e. the length to copy is taken from the length field referenced by the data offset, augmented by the additional_bytes field in the case of Aggregators). When `data_offset + data_length` is greater than the size of the sample, the bytes from the byte pointed to by `data_offset` until the end of the sample, inclusive, are copied. Resolution of an extractor may result in a reconstructed payload for which there are fewer bytes than what is indicated in the `NALUnitLength` of the first NAL in that reconstructed payload. In such cases, readers shall assume that only a single NAL unit was reconstructed by the extractors, and shall rewrite the `NALUnitLength` of that NAL to the appropriate value (i.e, size of the reconstructed payload minus (`LengthSizeMinusOne` + 1)).

NOTE   If the two tracks use different `lengthSizeMinusOne` values, then the extracted data will need re-formatting to conform to the destination track's length field size.

A.4   NAL unit header values for SVC

Both extractors and aggregators use NAL unit headers with the NAL unit header SVC extension. The NAL units extracted by an extractor or aggregated by an aggregator are all those NAL units that are referenced or included by recursively inspecting the contents of aggregators or extractors.

The fields `nal_ref_idc`, `idr_flag`, `priority_id`, `temporal_id`, `no_inter_layer_pred_flag`, `dependency_id`, `quality_id`, `temporal_id`, `use_ref_base_pic_flag`, `discardable_flag`, `output_flag`, `use_ref_base_pic_flag`, and `no_inter_layer_pred_flag` shall take the following values:

— `nal_ref_idc` shall be set to the highest value of the field in all the extracted or aggregated NAL units.

— `idr_flag` shall be set to the highest value of the field in all the extracted or aggregated NAL units.

— `priority_id`, `temporal_id`, `dependency_id`, and `quality_id` shall be set to the lowest values of the fields, respectively, in all the extracted or aggregated NAL units.

— `discardable_flag` shall be set to 1 if and only if all the extracted or aggregated NAL units have the `discardable_flag` set to 1, and set to 0 otherwise.

— `output_flag` should be set to 1 if at least one of the aggregated or extracted NAL units has this flag set to 1, and otherwise set to 0.

— `use_ref_base_pic_flag` shall be set to 1 if and only if at least one of the extracted or aggregated VCL NAL units have the `use_ref_base_pic_flag` set to 1, and set to 0 otherwise.

— `no_inter_layer_pred_flag` shall be set to 1 if and only if all the extracted or aggregated VCL NAL units have the `no_inter_layer_pred_flag` set to 1, and set to 0 otherwise.

If the set of extracted or aggregated NAL units is empty, then each of these fields takes a value conformant with the mapped tier description.

NOTE 1   Aggregators can group NAL units with different scalability information.

NOTE 2   Aggregators can be used to group NAL units belonging to a level of scalability that is not signalled by the NAL unit header SVC extension (e.g. NAL units belonging to a region of interest). The description of such Aggregators can be done with the tier description and the NAL unit map groups. In this case more than one Aggregator with the same scalability information can occur in one sample.

NOTE 3   If multiple scalable tracks reference the same media data, then an aggregator should group NAL units with identical scalability information only. This ensures that the resulting pattern can be accessed by each of the tracks.

NOTE 4   If no NAL unit of a particular layer exists in an access unit then an empty Aggregator (in which the length of the Aggregator includes only the header, and additional_bytes is zero) can exist.

A.5   NAL unit header values for MVC and MVC+D depth NAL units

Both Aggregators and Extractors use the NAL unit header MVC extension. The NAL units extracted by an extractor or aggregated by an aggregator are all those NAL units that are referenced or included by recursively inspecting the contents of aggregators or extractors.

The fields `nal_ref_idc`, `non_idr_flag`, `priority_id`, `view_id`, `temporal_id`, `anchor_pic_flag`, and `inter_view_flag` shall take the following values:

— `nal_ref_idc` shall be set to the highest value of the field in all the aggregated or extracted NAL units.

— `non_idr_flag` shall be set to the lowest value of the field in all the aggregated or extracted NAL units.

— `priority_id` and `temporal_id` shall be set to the lowest values of the fields, respectively, in all the aggregated or extracted NAL units.

— `view_id` shall be set to the view_id value of the VCL NAL unit with the lowest view order index among all the aggregated or extracted VCL NAL units.

— `anchor_pic_flag` and `inter_view_flag` shall be set to the highest value of the fields, respectively, in all the aggregated or extracted VCL NAL units.

A.6 NAL unit header values for 3D-AVC NAL units

Both Aggregators and Extractors use the NAL unit header 3D-AVC extension specified in ISO/IEC 14496-10:2014, Annex J. The NAL units extracted by an extractor or aggregated by an aggregator are all those NAL units that are referenced or included by recursively inspecting the contents of aggregators or extractors. All the aggregated or extracted VCL NAL units shall have nal_unit_type equal to 21 and avc_3d_extension_flag equal to 1.

The fields nal_ref_idc, view_idx, depth_flag, non_idr_flag, temporal_id, anchor_pic_flag, and inter_view_flag shall take the following values:

— nal_ref_idc shall be set to the highest value of the field in all the aggregated or extracted NAL units.

— view_idx shall be set to the lowest view order index among all the aggregated or extracted VCL NAL units.

— depth_flag shall be set to the lowest depth_flag value among all the aggregated or extracted VCL NAL units.

— non_idr_flag shall be set to the lowest value of the field in all the aggregated or extracted VCL NAL units.

— temporal_id shall be set to the lowest value of the field in all the aggregated or extracted VCL NAL units.

— anchor_pic_flag and inter_view_flag shall be set to the highest value of the fields, respectively, in all the aggregated or extracted VCL NAL units.

A.7 Extractors for HEVC and L-HEVC tracks

A.7.1 Definition

This subclause describes extractors for HEVC and L-HEVC tracks. Extractors enable compact formation of tracks that extract NAL unit data by reference.

An extractor contains one or more constructors. The following constructors are specified:

a) A sample constructor extracts, by reference, NAL unit data from a sample of another track.

b) An in-line constructor includes NAL unit data.

An aggregator may include or reference extractors. An extractor may reference aggregators. When an extractor is processed by a file reader that requires it, the extractor is logically replaced by the bytes resulting when resolving the contained constructors in their appearance order.

The bytes referred to by a sample constructor shall not contain extractors; an extractor shall not reference, directly or indirectly, another extractor.

NOTE   The track that is referenced can contain extractors even though the data that is referenced by the extractor shall not.

An extractor may contain one or more constructors for extracting data from the current track or from another track that is linked to the track in which the extractor resides by means of a track reference of type 'scal'.

The bytes of a resolved extractor shall be one of the following:

a) One entire NAL unit; note that when an Aggregator is referenced, both the included and referenced bytes are copied

b) More than one entire NAL unit

In both cases the bytes of the resolved extractor start with a valid length field and a NAL unit header.

The bytes of a sample constructor are copied only from the single identified sample in the track referenced through the indicated `'scal'` track reference. The alignment is on decoding time, i.e. using the time-to-sample table only, followed by a counted offset in sample number. Extractors are a media-level concept and hence apply to the destination track before any edit list is considered. (However, one would normally expect that the edit lists in the two tracks would be identical).

A.7.2 Syntax

```
class aligned(8) Extractor () {

   NALUnitHeader();

   do {

      unsigned int(8)   constructor_type;

      if( constructor_type == 0 )

         SampleConstructor();

      else if( constructor_type == 2 )

         InlineConstructor();

   } while( !EndOfNALUnit() )

}
```

NOTE   The syntax of Extractors does not always follow the NAL unit syntax and the NAL unit constraints specified in ISO/IEC 23008-2. For example, there can be three continuous bytes equal to a value in the range of 0x000000 to 0x000010, inclusive. This speciﬁation disallows the presence of Extractors in a video bitstream output from parsing a file, therefore formal non-compliance with the video specifications is immaterial as they will never be presented to a video decoder.

A.7.3 Semantics

`NALUnitHeader()`: The first two bytes of ISO/IEC 23008-2 NAL units.

`nal_unit_type` shall be set to 49 for ISO/IEC 23008-2 video. `forbidden_zero_bit` shall be set as specified in ISO/IEC 23008-2. Other fields (`nuh_layer_id` and `nuh_temporal_id_plus1`) shall be set as specified in A.8.

`constructor_type` specifies the constructor that follows. `SampleConstructor` and `InlineConstructor` correspond to `constructor_type` equal to 0 and 2, respectively. Other values of `constructor_type` are reserved.

`EndOfNALUnit()` is a function that returns 0 (false) when more data follows in this extractor; otherwise it returns 1 (true).

A.7.4 Sample constructor

A.7.4.1 Syntax

```
class aligned(8) SampleConstructor () {

   unsigned int(8) track_ref_index;

   signed int(8) sample_offset;

   unsigned int((lengthSizeMinusOne+1)*8)

      data_offset;
```

```
    unsigned int((lengthSizeMinusOne+1)*8)

        data_length;

}
```

A.7.4.2   Semantics

`track_ref_index`: as specified in A.3.3.

`sample_offset`: as specified in A.3.3.

`data_offset`: The offset of the first byte within the reference sample to copy. If the extraction starts with the first byte of data in that sample, the offset takes the value 0.

`data_length`: The number of bytes to copy. If this field takes the value 0, `data_offset` shall refer to the beginning of a NAL unit length field and the entire single referenced NAL unit is copied (i.e. the length to copy is taken from the length field referenced by `data_offset`, augmented by the additional_bytes field in the case of Aggregators). When `data_offset` + `data_length` is greater than the size of the sample, the bytes from the byte pointed to by `data_offset` until the end of the sample, inclusive, are copied, i.e. `data_length` is resolved as (`sample_size` - `data_offset`). Resolution of an extractor may result in a reconstructed payload for which there are fewer bytes than what is indicated in the `NALUnitLength` of the first NAL in that reconstructed payload. In such cases, readers shall assume that only a single NAL unit was reconstructed by the extractors, and shall rewrite the `NALUnitLength` of that NAL to the appropriate value (i.e, size of the reconstructed payload minus (`LengthSizeMinusOne` + 1)).

NOTE   If the two tracks use different `lengthSizeMinusOne` values, then the extracted data will need re-formatting to conform to the destination track's length field size.

A.7.5   In-line constructor

A.7.5.1   Syntax

```
class aligned(8) InlineConstructor () {

    unsigned int(8) length;

    unsigned int(8) inline_data[length];

}
```

A.7.5.2   Semantics

`length`: the number of bytes that belong to the `InlineConstructor` following this field. The value of `length` shall be greater than 0. The value of `length` equal to 0 is reserved.

`inline_data`: the data bytes to be returned when resolving the in-line constructor.

A.8   NAL unit header values for ISO/IEC 23008-2

Both Aggregators and Extractors use the NAL unit header as specified in ISO/IEC 23008-2. The NAL units extracted by an extractor or aggregated by an aggregator are all those NAL units that are referenced or included by recursively inspecting the contents of aggregators or extractors.

The fields `nuh_layer_id` and `nuh_temporal_id_plus1` shall be set as follows:

—   `nuh_layer_id` shall be set to the lowest value of the field in all the aggregated or extracted NAL units.

—   `nuh_temporal_id_plus1` shall be set to the lowest value of the field in all the aggregated or extracted NAL units.

Annex B
(normative)

SVC, MVC, and MVD sample group and sub-track definitions

B.1 General

The following sample groups may be used in an SVC, MVC, or MVD track to document the structure of the SVC, MVC, or MVD stream and to ease obtaining information of subsets of the stream and extraction of any of the subsets.

If views from the same MVC or MVD bitstream are stored in multiple MVC or MVD tracks and one or more of these tracks contain multiple views, sample group entries and map groups can be used for these tracks containing multiple views.

There are a number of boxes, defined below, which may occur in the sample group description, namely the Scalable Group Entry for an SVC stream or the Multiview Group Entry for an MVC or MVD stream.

Each Scalable Group Entry or Multiview Group Entry documents a subset of the SVC stream or the MVC or MVD stream, respectively. Each of the subsets is associated with a tier and may contain one or more operating points. A grouping type of `'scif'` or `'mvif'` is used to define Scalable Group Entries or Multiview Group Entries, respectively.

For each tier, there may be more than one Scalable Group Entry or Multiview Group Entry in the SampleGroupDescriptionBox of grouping type `'scif'` or `'mvif'`, respectively. Only one of those entries is the primary definition of the tier.

Though the Scalable and Multiview Group Entries are contained in the SampleGroupDescription box, the grouping is not a true *sample* grouping as each sample may be associated with more than one tier, as these groups are used to describe *sections* of the samples – the NAL units. As a result, it is possible that there may not be a SampleToGroup box of the grouping type `'scif'` or `'mvif'`, unless it happens that a group does, in fact, describe a whole sample. Even if a SampleToGroup box of the grouping type `'scif'` or `'mvif'` is present, the information is not needed for extraction of NAL units of tiers; instead, the map groups shall always document the 'pattern' of NAL units within the samples and provide the NAL-unit-to-tier mapping information that may be needed for extraction of NAL units.

A multiview group specifies an MVC or MVD operating point and is therefore associated with the target output views of the MVC or MVD operating point. The Multiview Group box (7.7.3), is used to specify a multiview group. Many of the boxes used to characterize SVC, MVC, and MVD tiers are also used to characterize MVC or MVD operating points and can therefore be contained in the Multiview Group box too.

B.2   Definition

B.2.1   Tier information box

B.2.1.1   Definition

Box Type: 'tiri'

Container: ScalableGroupEntry or MultiviewGroupEntry or MultiviewGroupBox

Mandatory: Yes

Quantity: Zero or One // depends on primary_definition

The tier information box provides information about the profile, level, frame size, discardability, and frame-rate of a covered bitstream subset. If the Tier Information box is included in a Scalable Group entry or a Multiview Group entry, the covered bitstream subset consists of the tier and tiers it depends upon. If the Tier Information box is included in a Multiview Group box, the covered bitstream subset consists of the target output views of the multiview group and all the views required for decoding the target output views.

B.2.1.2   Syntax

```
class TierInfoBox extends Box('tiri'){ //Mandatory Box

   unsigned int(16) tierID;

   unsigned int(8) profileIndication;

   unsigned int(8) profile_compatibility;

   unsigned int(8) levelIndication;

   bit(8) reserved = 0;


   unsigned int(16) visualWidth;

   unsigned int(16) visualHeight;


   unsigned int(2) discardable;

   unsigned int(2) constantFrameRate;

   bit(4) reserved = 0;

   unsigned int(16) frameRate;

}
```

B.2.1.3   Semantics

`tierID` gives the identifier of the tier, when the Tier Information box is included a Scalable Group entry or a Multiview Group entry. Otherwise, the semantics of `tierID` are unspecified, and in this case, `tierID` shall be set to the reserved value 0.

`profileIndication` contains the profile_idc as defined in ISO/IEC 14496-10, when the parameter applies to the covered bitstream subset.

`profile_compatibility` is a byte defined exactly the same as the byte that occurs between the profile_idc and level_idc in a sequence parameter set or a subset sequence parameter set, as defined in ISO/IEC 14496-10:2014 Annex G, H or I, when the parameters apply to the covered bitstream subset.

`levelIndication` contains the level_idc as defined in ISO/IEC 14496-10, when the parameter applies to the covered bitstream subset. If the Tier Information Box is included in a Multiview Group Entry, `levelIndication` shall be valid when all the views of the covered bitstream subset are target output views. If the Tier Information Box is included in a Multiview Group Box, `levelIndication` shall be valid when the views specified by the respective multiview group are the target output views. If `levelIndication` is equal to 0 for an MVC or MVD stream, the level that applies to the covered bitstream subset and operating with all the views being target output views is unspecified.

The profile, profile compatibility flags and level indicated by the fields `profileIndication`, `profile_compatibility`, and `levelIndication` specifies an interoperability point with which the covered bitstream subset, and, for MVC or MVD, operating with the target output views as specified in the semantics of `levelIndication`, is compatible.

`visualWidth` gives the value of the width of the coded picture (of an SVC stream), coded sub-picture (of an SVC stream), or coded view component (of an MVC or MVD stream) in luma pixels of the representation of this tier in the stream or any view component of the covered bitstream subset. A coded sub-picture consists of a proper subset of coded slices of a coded picture. A tier may consist of only sub-pictures. In this case, the tier is referred to as a sub-picture tier. A sub-picture tier may represent a region-of-interest part of the region represented by the entire stream.

NOTE   The tier representation of a sub-picture tier might not be a valid stream. One example is as follows. An AVC bitstream is encoded using two slice groups. The first slice group includes the macroblocks representing a region-of-interest and is coded without referring to slices in the other slice group for inter prediction over all the access units. The slices of the first slice group in each access unit then form a sub-picture and a sub-picture tier can be specified to include all the sub-pictures over all the access units.

`visualHeight` gives the value of the height of the coded picture (of an SVC stream), coded sub-picture (of an SVC stream), or coded view component (of an MVC or MVD stream) in luma pixels of the representation of this tier in the stream or any view component of the covered bitstream subset.

`discardable` takes one of the following values; the value 02 is reserved.

— 00 this tier does not contain NAL units with discardable_flag (for SVC) equal to 1 or inter_view_flag (for MVC or MVD) equal to 0.

— 01 this tier contains both NAL units with discardable_flag (for SVC) equal to 1 or inter_view_flag (for MVC or MVD) equal to 0 and discardable_flag (for SVC) equal to 0 or inter_view_flag (for MVC or MVD) equal to 1.

— 03 all NAL units in this tier are with discardable_flag (for SVC) equal to 1 or inter_view_flag (for MVC or MVD) equal to 0.

`constantFrameRate` specifies if the frame rate of this tier is constant. A value of 0 denotes a non-constant frame rate, a value of 1 denotes a constant frame rate and a value of 2 denotes that it is not clear whether the frame rate is constant. A value of 3 is reserved.

`frameRate` gives the frame rate when the bitstream corresponding to this tier and all the lower tiers that this tier depends on is decoded in frames per second rounded to the closest integer using the Round function specified in ISO/IEC 14496-10. If `constantFrameRate` has a value of 0 or 2 then `frameRate` gives the average frame rate. If `constantFrameRate` has a value of 1 then `frameRate` gives the constant frame rate. `frameRate` equal to 0 indicates an unspecified frame rate. For SVC streams, decoded frames, complementary field pairs and non-paired fields are regarded as frames when deriving the value of frameRate. For MVC or MVD treams, decoded view components of any single view only are regarded as frames when deriving the value of `frameRate`, regardless of the total number of the views, since all output views are required to have simultaneous view components.

B.2.2 Tier bit rate box

B.2.2.1 Definition

Box Type: 'tibr'

Container: ScalableGroupEntry or MultiviewGroupEntry or MultiviewGroupBox

Mandatory: No

Quantity: Zero or One

When included in a Scalable Group entry or a Multiview Group entry, the tier bit rate box provides information about the bit rate values of a tier. Two sets of information are provided: for the tier representation, including all the tiers on which the current tier depends, and for the tier alone. Similarly, for each set of information, the following values are supplied:

— For SVC streams, the lowest long-term average bit rate that this tier could deliver. Let maxDid be the greatest dependency_id for all NAL units of the tier, and minQid be the least quality_id for all the NAL units of the tier and having dependency_id equal to maxDid. The following NAL units of this tier are not considered in calculating this bit rate value: those having dependency_id equal to maxDid and quality_id greater than minQid.

— For MVC or MVD streams, the lowest long-term average bit rate that this tier could deliver is equal to the long-term average bit rate of the tier, when all NAL units of the tier are considered.

— Tthe long-term average bit rate of the tier; all NAL units of the tier are considered.

— The maximum, or peak, bit rate of the tier; all NAL units of the tier are considered.

When included in a Multiview Group box, the tier bit rate box provides information about the bit rate values of the covered bitstream subset consisting of the target output views indicated by the multiview group and all the views required for decoding of the target output views. The maximum and long-term average bit rate for the covered bitstream subset are provided.

B.2.2.2 Syntax

```
class TierBitRateBox extends Box('tibr'){

    unsigned int(32) baseBitRate;

    unsigned int(32) maxBitRate;

    unsigned int(32) avgBitRate;


    unsigned int(32) tierBaseBitRate;

    unsigned int(32) tierMaxBitRate;

    unsigned int(32) tierAvgBitRate;

}
```

B.2.2.3 Semantics

baseBitRate gives the lowest long-term average bit rate in bits/second of the stream made from this tier and the lower tiers this tier depends on over the entire stream.

For SVC streams, `baseBitRate` is derived as follows. Let maxDid be the greatest dependency_id for all NAL units of the tier, and minQid be the least quality_id for all NAL units of the tier and having dependency_id equal to maxDid. The NAL units that are taken into account when calculating this bit rate value are as follows: 1) all NAL units of the tier except for those having dependency_id equal to maxDid and quality_id greater than minQid; 2) all NAL units of the lower tiers the current tier depends on.

For MVC or MVD streams, `baseBitRate` shall be equal to `avgBitRate`.

`maxBitRate` gives the maximum bit rate in bits/second of the stream containing all NAL unit mapped to this tier and the lower tiers this tier depends on, over any window of one second. All NAL units in this tier and the lower tiers this tier depends on are taken into account.

`avgBitRate` gives the long-term average bit rate in bits/second of the stream containing all NAL unit mapped to this tier and the lower tiers this tier depends on, averaged over the entire stream. All NAL units in this tier and the lower tiers this tier depends on are taken into account.

`tierBaseBitRate` gives the lowest long-term average bit rate in bits/second of the stream made from only this tier over the entire stream. For SVC streams, the set of NAL units that are taken into account when calculating this bit rate value is the same as for baseBitRate but excluding all NAL units of the lower tiers this tier depends on. For MVC or MVD streams, `tierBaseBitRate` shall be equal to `tierAvgBitRate`.

`tierMaxBitRate` gives the maximum bit rate in bits/second that is provided by only this tier over any window of one second. All NAL units mapped to this tier are taken into account. All NAL units of the lower tiers this tier depends on are not considered.

`tierAvgBitRate` - gives the long-term average bit rate in bits/second that is provided by only this tier, averaged over the entire stream. All NAL units mapped to this tier are taken into account. All NAL units of the lower tiers this tier depends on are not considered.

B.2.3   Priority range

B.2.3.1   Definition

Box Type: `'svpr'`

Container: ScalableGroupEntry or MultiviewGroupEntry

Mandatory: Yes

Quantity: Exactly One

NOTE   This box was previously called SVCPriorityRangeBox.

This box reports the minimum and maximum priority_id of the NAL units mapped to this tier.

B.2.3.2   Syntax

```
class PriorityRangeBox extends Box('svpr') {

    bit(2) reserved1 = 0;

    unsigned int(6) min_priorityId;

    bit(2) reserved2 = 0;

    unsigned int(6) max_priorityId;

}
```

B.2.3.3 Semantics

`min_priority_id`, `min_priority_id` take the minimum or maximum value of the priority_id syntax element that is present in the NAL unit header extension of the SVC, MVC or MVC+D depth NAL units mapped to the tier. For AVC streams this takes the value that is, or would be, in the prefix NAL unit.

B.2.4 SVC dependency range

B.2.4.1 Definition

Box Types: `'svdr'`

Container: ScalableGroupEntry

Mandatory: Yes

Quantity: Exactly One

This box reports the minimum and maximum dependency_id of the NAL units mapped to this tier.

The field min_temporal_id reports the minimum value of temporal_id of the NAL units in the tier having dependency_id equal to min_dependency_id, Similarly the field min_quality_id reports the minimum quality_id of those NAL units. The fields max_temporal_id and max_quality_id similarly report on the maximum values of the respective fields in those NAL units having dependency_id equal to max_dependency_id.

B.2.4.2 Syntax

```
class SVCDependencyRangeBox extends Box('svdr') {

    unsigned int(3) min_dependency_id;

    unsigned int(3) min_temporal_id;

    bit(6) reserved = 0;

    unsigned int(4) min_quality_id;

    unsigned int(3) max_dependency_id;

    unsigned int(3) max_temporal_id;

    bit(6) reserved = 0;

    unsigned int(4) max_quality_id;

}
```

B.2.4.3 Semantics

`min_dependency_id`, `max_dependency_id` take the minimum or maximum value of the dependency_id syntax element that is present in the scalable extension NAL unit header defined in the SVC video specification of the NAL units mapped to the tier. For AVC streams this takes the value that is, or would be, in the prefix NAL unit (note: this value is zero).

`min_temporal_id`, `max_temporal_id` take the minimum or value of the temporal_id syntax element that is present in the scalable extension NAL unit header defined in the SVC video specification of the NAL units mapped to the tier having dependency_id equal to `min_dependency_id` and `max_dependency_id` respectively . For AVC streams this takes the value that is, or would be, in the prefix NAL unit.

min_quality_id, max_quality_id take the minimum or value of the quality_id syntax element that is present in the scalable extension NAL unit header defined in the SVC video specification of the NAL units mapped to the tier having dependency_id equal to min_dependency_id and max_dependency_id respectively . For AVC streams this takes the value that is, or would be, in the prefix NAL unit.

B.2.5   Initial parameter sets box

B.2.5.1   Definition

Box Type: 'svip'

Container: ScalableGroupEntry or MultiviewGroupEntry

Mandatory: No

Quantity: Zero or One

The initial parameter sets box documents which parameter sets are needed for decoding this tier and all the lower tiers it depends on.

B.2.5.2   Syntax

```
class InitialParameterSetBox extends Box ('svip') {

   unsigned int(8) sps_id_count;

   for (i=0; i< sps_id_count; i++)

      unsigned int(8) SPS_index;

   unsigned int(8) pps_id_count;

   for (i=0; i< pps_id_count; i++)

      unsigned int(8) PPS_index;

}
```

B.2.5.3   Semantics

sps_id_count, pps_id_count gives the number of entries in the following tables.

SPS_index specifies that the SPS or subset SPS with this index is needed for decoding this tier and all the lower tiers it depends on. These are 1-based indices into the arrays in SVCDecoderConfigurationRecord, MVCDecoderConfigurationRecord, or MVDDecoderConfigurationRecord.

PPS_index specifies that the PPS with this index is needed for decoding this tier and all the lower tiers it depends on. These are 1-based indices into the arrays in SVCDecoderConfigurationRecord, MVCDecoderConfigurationRecord, or MVDDecoderConfigurationRecord.

B.2.6   SVC rect region box

B.2.6.1   Definition

Box Type: 'rrgn'

Container: ScalableGroupEntry

Mandatory: No

Quantity: Zero or One

The SVC rect region box documents the geometry information of the region represented by the current tier relative to the region represented by another tier. When extended spatial scalability was used to encode in the current tier a cropped region of another tier, then the geometry information of the cropped region can be signalled by this box. This box can also be used to signal the geometry information of a region-of-interest (ROI) when the current tier is a sub-picture tier. This area can either be static for all samples or vary at sample-by-sample basis. Note that it is possible that independent sub-pictures do not depend on all the tiers with lower tierID. In this case dependencies can be given with the tier dependency box.

B.2.6.2   Syntax

```
class RectRegionBox extends Box('rrgn'){

   unsigned int(16) base_region_tierID;

   unsigned int(1) dynamic_rect;

   bit(7) reserved = 0;

   if(dynamic_rect == 0) {

      unsigned int(16) horizontal_offset;

      unsigned int(16) vertical_offset;

      unsigned int(16) region_width;

      unsigned int(16) region_height;

   }

}
```

B.2.6.3   Semantics

base_region_tierID gives the tierId value of the tier wherein the represented region is used as the base region for derivation of the region represented by the current tier.

dynamic_rect equal to 1 indicates that the region represented by the current tier is a dynamically changing rectangular part of the base region. Otherwise the region represented by the current tier is a fixed rectangular part of the base region.

horizontal_offset and vertical_offset give respectively the horizontal and vertical offsets of the top-left pixel of the rectangular region represented by the tier, in relative to the top-left pixel of the base region, in luma samples of the base region.

region_width and region_height give respectively the width and height of the rectangular region represented by the tier, in luma samples of the base region.

B.2.7   Buffering information box

B.2.7.1   Definition

Box Type: `'buff'`

Container: ScalableGroupEntry or MultiviewGroupEntry or MultiviewGroupBox

Mandatory: No

Quantity: Zero or One

The BufferingBox contains the buffer information of the covered bitstream subset. If the Buffering box is included in a Scalable Group entry or a Multiview Group entry, the covered bitstream subset consists of the tier and all tiers on which it depends. If the Buffering box is included in a Multiview Group box, the covered bitstream subset consists of the target output views of the multiview group and all the views required for decoding the target output views.

B.2.7.2   Syntax

```
class BufferingBox extends Box('buff'){

   unsigned int(16)      operating_point_count

   for (i = 0; i < operating_point_count; i++){

      unsigned int (32)    byte_rate

      unsigned int (32)    cpb_size

      unsigned int (32)    dpb_size

      unsigned int (32)    init_cpb_delay

      unsigned int (32)    init_dpb_delay

   }

}
```

B.2.7.3   Semantics

`operating_point_count` specifies the number of HRD operating points for the covered bitstream subset. Values of the HRD parameters are specified separately for each operating point. The value of operating_point_count shall be greater than 0.

`byte_rate` specifies the input byte rate (in bytes per second) to the coded picture buffer (CPB) of the HRD. The covered bitstream subset is constrained by the value of BitRate equal to byte_rate * 8 for NAL HRD parameters as specified in ISO/IEC 14496-10. For VCL HRD parameters, the value of BitRate is equal to byte_rate * 40 / 6. The value of byte_rate shall be greater than 0.

`cpb_size` specifies the required size of the coded picture buffer in bytes. The covered bitstream subset is constrained by the value of CpbSize equal to cpb_size * 8 for NAL HRD parameters as specified in ISO/IEC 14496-10. For VCL HRD parameters, the value of CpbSize is equal to cpb_size * 40 / 6.

At least one pair of values of byte_rate and cpb_size of the same operating point shall conform to the maximum bit rate and CPB size allowed by profile and level of the covered bitstream subset.

`dpb_size` specifies the required size of the decoded picture buffer (DPB), in unit of bytes. The covered bitstream subset is constrained by the value of max_dec_frame_buffering equal to Min(16, Floor(