International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

H.222.0

(03/2017)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Infrastructure of audiovisual services – Transmission multiplexing and synchronization

---

Information technology – Generic coding of moving pictures and associated audio information: Systems

Recommendation ITU-T H.222.0

ITU-T

International Telecommunication Union

ITU-T H-SERIES RECOMMENDATIONS

AUDIOVISUAL AND MULTIMEDIA SYSTEMS

CHARACTERISTICS OF VISUAL TELEPHONE SYSTEMS	H.100–H.199
INFRASTRUCTURE OF AUDIOVISUAL SERVICES	
General	H.200–H.219
Transmission multiplexing and synchronization	**H.220–H.229**
Systems aspects	H.230–H.239
Communication procedures	H.240–H.259
Coding of moving video	H.260–H.279
Related systems aspects	H.280–H.299
Systems and terminal equipment for audiovisual services	H.300–H.349
Directory services architecture for audiovisual and multimedia services	H.350–H.359
Quality of service architecture for audiovisual and multimedia services	H.360–H.369
Telepresence	H.420–H.429
Supplementary services for multimedia	H.450–H.499
MOBILITY AND COLLABORATION PROCEDURES	
Overview of Mobility and Collaboration, definitions, protocols and procedures	H.500–H.509
Mobility for H-Series multimedia systems and services	H.510–H.519
Mobile multimedia collaboration applications and services	H.520–H.529
Security for mobile multimedia systems and services	H.530–H.539
Security for mobile multimedia collaboration applications and services	H.540–H.549
Mobility interworking procedures	H.550–H.559
Mobile multimedia collaboration inter-working procedures	H.560–H.569
BROADBAND, TRIPLE-PLAY AND ADVANCED MULTIMEDIA SERVICES	
Broadband multimedia services over VDSL	H.610–H.619
Advanced multimedia services and applications	H.620–H.629
Ubiquitous sensor network applications and Internet of Things	H.640–H.649
IPTV MULTIMEDIA SERVICES AND APPLICATIONS FOR IPTV	
General aspects	H.700–H.719
IPTV terminal devices	H.720–H.729
IPTV middleware	H.730–H.739
IPTV application event handling	H.740–H.749
IPTV metadata	H.750–H.759
IPTV multimedia application frameworks	H.760–H.769
IPTV service discovery up to consumption	H.770–H.779
Digital Signage	H.780–H.789
E-HEALTH MULTIMEDIA SERVICES AND APPLICATIONS	
Personal health systems	H.810–H.819
Interoperability compliance testing of personal health systems (HRN, PAN, LAN, TAN and WAN)	H.820–H.859
Multimedia e-health data exchange services	H.860–H.869

For further details, please refer to the list of ITU-T Recommendations.

INTERNATIONAL STANDARD ISO/IEC 13818-1
RECOMMENDATION ITU-T H.222.0

Information technology – Generic coding of moving pictures and associated audio information: Systems

Summary

This Recommendation | International Standard specifies the system layer of the coding. It was developed in 1994 to principally support the combination and synchronization of video and audio coding methods defined in ISO/IEC 13818 Part 2 (ITU-T H.262) and Part 3. Since 1994, this standard has been extended to support additional video coding specifications (e.g., ISO/IEC 14496-2, ITU-T H.264 | ISO/IEC 14496-10, ITU-T H.265 | ISO/IEC 23008-2 and ITU-T T.800 | ISO/IEC 15444-1 Annex M JPEG 2000 video), audio coding specifications (e.g., ISO/IEC 13818-7 and ISO/IEC 14496-3), system streams (e.g., ISO/IEC 14496-1 and ISO/IEC 15938-1), ISO/IEC 23009-1 dynamic adaptive streaming over HTTP (DASH), ISO/IEC 13818-11 intellectual property management and protection (IPMP) as well as generic metadata. The system layer supports six basic functions:

1) the synchronization of multiple compressed streams on decoding;
2) the interleaving of multiple compressed streams into a single stream;
3) the initialization of buffering for decoding start up;
4) continuous buffer management;
5) time identification; and
6) multiplexing and signalling of various components in a system stream.

Recommendation ITU-T H.222.0 | ISO/IEC 13818-1 multiplexed bit stream is either a transport stream or a program stream. Both streams are constructed from packetized elementary stream (PES) packets and packets containing other necessary information. Both stream types support multiplexing of video and audio compressed streams from one program with a common time base. The transport stream additionally supports the multiplexing of video and audio compressed streams from multiple programs with independent time bases. For almost error-free environments the program stream is generally more appropriate, supporting software processing of program information. The transport stream is more suitable for use in environments where errors are likely.

Recommendation ITU-T H.222.0 | ISO/IEC 13818-1 multiplexed bit stream, whether a transport stream or a program stream, is constructed in two layers: the outermost layer is the system layer, and the innermost is the compression layer. The system layer provides the functions necessary for using one or more compressed data streams in a system. The video and audio parts of this Specification define the compression coding layer for audio and video data. Coding of other types of data is not defined by this Recommendation | International Standard, but is supported by the system layer provided that the other types of data adhere to the constraints defined in this Recommendation | International Standard.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T H.222.0	1995-07-10	15	11.1002/1000/1071
1.1	ITU-T H.222.0 (1995) Amd. 1	1996-11-11	16	11.1002/1000/3834
1.2	ITU-T H.222.0 (1995) Amd. 2	1996-11-11	16	11.1002/1000/4096
1.3	ITU-T H.222.0 (1995) Technical Cor. 1	1998-02-06	16	11.1002/1000/4532
1.4	ITU-T H.222.0 (1995) Amd. 3	1998-02-06	16	11.1002/1000/4228
1.5	ITU-T H.222.0 (1995) Amd. 4	1998-02-06	16	11.1002/1000/4229
1.6	ITU-T H.222.0 (1995) Amd. 5	1999-05-27	16	11.1002/1000/4498
1.7	ITU-T H.222.0 (1995) Amd. 6	1999-05-27	16	11.1002/1000/4671

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

2.0	ITU-T H.222.0	2000-02-17	16	11.1002/1000/5142
2.1	ITU-T H.222.0 (2000) Technical Cor. 1	2001-03-01	16	11.1002/1000/5419
2.2	ITU-T H.222.0 (2000) Technical Cor. 2	2002-03-29	16	11.1002/1000/5675
2.3	ITU-T H.222.0 (2000) Amd. 1	2002-12-14	16	11.1002/1000/6190
2.4	ITU-T H.222.0 (2000) Amd. 1/Cor. 1	2003-06-29	16	11.1002/1000/6449
2.5	ITU-T H.222.0 (2000) Amd. 2	2003-06-29	16	11.1002/1000/6363
2.6	ITU-T H.222.0 (2000) Amd. 3	2004-03-15	16	11.1002/1000/7208
2.7	ITU-T H.222.0 (2000) Technical Cor. 3	2005-01-08	16	11.1002/1000/7435
2.8	ITU-T H.222.0 (2000) Amd. 4	2005-01-08	16	11.1002/1000/7436
2.9	ITU-T H.222.0 (2000) Amd. 5	2005-01-08	16	11.1002/1000/7437
2.10	ITU-T H.222.0 (2000) Technical Cor. 4	2005-09-13	16	11.1002/1000/8560
3.0	ITU-T H.222.0	2006-05-29	16	11.1002/1000/8802
3.1	ITU-T H.222.0 (2006) Amd. 1	2007-01-13	16	11.1002/1000/9024
3.2	ITU-T H.222.0 (2006) Amd. 2	2007-08-29	16	11.1002/1000/9214
3.3	ITU-T H.222.0 (2006) Cor. 1	2008-06-13	16	11.1002/1000/9471
3.4	ITU-T H.222.0 (2006) Cor. 2	2009-03-16	16	11.1002/1000/9692
3.5	ITU-T H.222.0 (2006) Amd. 3	2009-03-16	16	11.1002/1000/9691
3.6	ITU-T H.222.0 (2006) Cor. 3	2009-12-14	16	11.1002/1000/10621
3.7	ITU-T H.222.0 (2006) Cor. 4	2009-12-14	16	11.1002/1000/10622
3.8	ITU-T H.222.0 (2006) Amd. 4	2009-12-14	16	11.1002/1000/10623
3.9	ITU-T H.222.0 (2006) Amd. 5	2011-05-14	16	11.1002/1000/11287
3.10	ITU-T H.222.0 (2006) Amd. 6	2011-05-14	16	11.1002/1000/11288
4.0	ITU-T H.222.0	2012-06-29	16	11.1002/1000/11655
4.1	ITU-T H.222.0 (2012) Amd. 1	2014-01-13	16	11.1002/1000/12054
4.2	ITU-T H.222.0 (2012) Amd. 2	2014-01-13	16	11.1002/1000/12055
4.3	ITU-T H.222.0 (2012) Amd. 3	2014-01-13	16	11.1002/1000/12056
4.4	ITU-T H.222.0 (2012) Amd. 4	2014-01-13	16	11.1002/1000/12057
4.5	ITU-T H.222.0 (2012) Amd. 5	2014-10-14	16	11.1002/1000/12306
5.0	ITU-T H.222.0	2014-10-14	16	11.1002/1000/12359
5.1	ITU-T H.222.0 (2014) Amd. 1	2015-04-29	16	11.1002/1000/12452
5.2	ITU-T H.222.0 (2014) Amd. 1 Cor. 1	2015-11-29	16	11.1002/1000/12625
5.3	ITU-T H.222.0 (2014) Amd. 2	2015-12-14	16	11.1002/1000/12632
5.4	ITU-T H.222.0 (2014) Amd. 3	2015-12-14	16	11.1002/1000/12633
5.5	ITU-T H.222.0 (2014) Amd. 1 Cor. 2	2016-07-14	16	11.1002/1000/12899
5.5	ITU-T H.222.0 (2014) Cor. 1	2016-07-14	16	11.1002/1000/12903
5.7	ITU-T H.222.0 (2014) Amd. 4	2016-07-14	16	11.1002/1000/12900
5.8	ITU-T H.222.0 (2014) Amd. 5	2016-07-14	16	11.1002/1000/12901
5.9	ITU-T H.222.0 (2014) Amd. 6	2016-07-14	16	11.1002/1000/12902
5.10	ITU-T H.222.0 (2014) Amd. 3 Cor. 1	2017-03-01	16	11.1002/1000/13184
5.10	ITU-T H.222.0 (2014) Cor. 2	2017-03-01	16	11.1002/1000/13188
5.12	ITU-T H.222.0 (2014) Amd. 7	2017-03-01	16	11.1002/1000/13186
5.13	ITU-T H.222.0 (2014) Amd. 8	2017-03-01	16	11.1002/1000/13187
6.0	ITU-T H.222.0	2017-03-01	16	11.1002/1000/13269

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Introduction

The systems part of this Recommendation | International Standard addresses the combining of one or more elementary streams of video and audio, as well as other data, into single or multiple streams which are suitable for storage or transmission. Systems coding follows the syntactical and semantic rules imposed by this Specification and provides information to enable synchronized decoding of decoder buffers over a wide range of retrieval or receipt conditions.

System coding shall be specified in two forms: the transport stream and the program stream. Each is optimized for a different set of applications. Both the transport stream and program stream defined in this Recommendation | International Standard provide coding syntax which is necessary and sufficient to synchronize the decoding and presentation of the video and audio information, while ensuring that data buffers in the decoders do not overflow or underflow. Information is coded in the syntax using time stamps concerning the decoding and presentation of coded audio and visual data and time stamps concerning the delivery of the data stream itself. Both stream definitions are packet-oriented multiplexes.

The basic multiplexing approach for single video and audio elementary streams is illustrated in Figure Intro. 1. The video and audio data is encoded as described in Rec. ITU-T H.262 | ISO/IEC 13818-2 and ISO/IEC 13818-3. The resulting compressed elementary streams are packetized to produce PES packets. Information needed to use PES packets independently of either transport streams or program streams may be added when PES packets are formed. This information is not needed and need not be added when PES packets are further combined with system level information to form transport streams or program streams. This systems standard covers those processes to the right of the vertical dashed line.

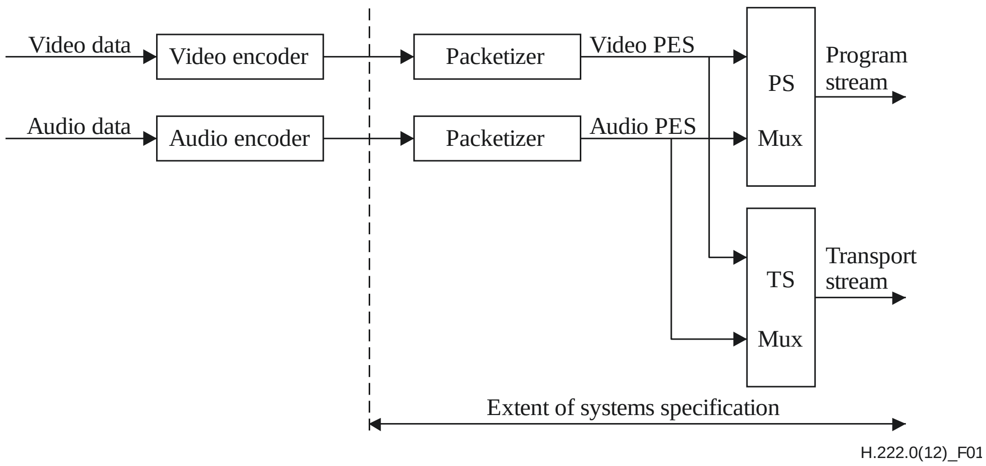


Figure Intro. 1 – Simplified overview of the scope of this Recommendation | International Standard

The program stream is analogous and similar to the ISO/IEC 11172 systems layer. It results from combining one or more streams of PES packets, which have a common time base, into a single stream.

For applications that require the elementary streams that comprise a single program to be in separate streams that are not multiplexed, the elementary streams can also be encoded as separate program streams, one per elementary stream, with a common time base. In this case the values encoded in the SCR fields of the various streams shall be consistent.

Like the single program stream, all elementary streams can be decoded with synchronization.

The program stream is designed for use in relatively error-free environments and is suitable for applications which may involve software processing of system information such as interactive multi-media applications. Program stream packets may be of variable and relatively great length.

The transport stream combines one or more programs with one or more independent time bases into a single stream. PES packets made up of elementary streams that form a program share a common timebase. The transport stream is designed for use in environments where errors are likely, such as storage or transmission in lossy or noisy media. Transport stream packets are 188 bytes in length.

Program and transport streams are designed for different applications and their definitions do not strictly follow a layered model. It is possible and reasonable to convert from one to the other; however, one is not a subset or superset of the other. In particular, extracting the contents of a program from a transport stream and creating a valid program stream is possible and is accomplished through the common interchange format of PES packets, but not all of the fields needed in a program stream are contained within the transport stream; some must be derived. The transport stream may be used to span a range of layers in a layered model, and is designed for efficiency and ease of implementation in high bandwidth applications.

The scope of syntactical and semantic rules set forth in the systems specification differs: the syntactical rules apply to systems layer coding only, and do not extend to the compression layer coding of the video and audio specifications; by contrast, the semantic rules apply to the combined stream in its entirety.

The systems specification does not specify the architecture or implementation of encoders or decoders, nor those of multiplexors or demultiplexors. However, bit stream properties do impose functional and performance requirements on encoders, decoders, multiplexors and demultiplexors. For instance, encoders must meet minimum clock tolerance requirements. Notwithstanding this and other requirements, a considerable degree of freedom exists in the design and implementation of encoders, decoders, multiplexors, and demultiplexors.

Intro. 1 Transport stream

The transport stream is a stream definition which is tailored for communicating or storing one or more programs of coded data according to Rec. ITU-T H.262 | ISO/IEC 13818-2 and ISO/IEC 13818-3 and other data in environments in which significant errors may occur. Such errors may be manifested as bit value errors or loss of packets.

Transport streams may be either fixed or variable rate. In either case the constituent elementary streams may either be fixed or variable rate. The syntax and semantic constraints on the stream are identical in each of these cases. The transport stream rate is defined by the values and locations of program clock reference (PCR) fields, which in general are separate PCR fields for each program.

There are some difficulties with constructing and delivering a transport stream containing multiple programs with independent time bases such that the overall bit rate is variable. Refer to 2.4.2.3.

The transport stream may be constructed by any method that results in a valid stream. It is possible to construct transport streams containing one or more programs from elementary coded data streams, from program streams, or from other transport streams which may themselves contain one or more programs.

The transport stream is designed in such a way that several operations on a transport stream are possible with minimum effort. Among these are:

1) Retrieve the coded data from one program within the transport stream, decode it and present the decoded results as shown in Figure Intro. 2.
2) Extract the transport stream packets from one program within the transport stream and produce as output a different transport stream with only that one program as shown in Figure Intro. 3.
3) Extract the transport stream packets of one or more programs from one or more transport streams and produce as output a different transport stream (not illustrated).
4) Extract the contents of one program from the transport stream and produce as output a program stream containing that one program as shown in Figure Intro. 4.
5) Take a program stream, convert it into a transport stream to carry it over a lossy environment, and then recover a valid, and in certain cases, identical program stream.

Figure Intro. 2 and Figure Intro. 3 illustrate prototypical demultiplexing and decoding systems which take as input a transport stream. Figure Intro. 2 illustrates the first case, where a transport stream is directly demultiplexed and decoded. Transport streams are constructed in two layers:

– a system layer; and
– a compression layer.

The input stream to the transport stream decoder has a system layer wrapped about a compression layer. Input streams to the video and audio decoders have only the compression layer.

Operations performed by the prototypical decoder which accepts transport streams either apply to the entire transport stream ("multiplex-wide operations"), or to individual elementary streams ("stream-specific operations"). The transport stream system layer is divided into two sub-layers, one for multiplex-wide operations (the transport stream packet layer), and one for stream-specific operations (the PES packet layer).

A prototypical decoder for transport streams, including audio and video, is also depicted in Figure Intro. 2 to illustrate the function of a decoder. The architecture is not unique – some system decoder functions, such as decoder timing control, might equally well be distributed among elementary stream decoders and the channel-specific decoder – but this figure is useful for discussion. Likewise, indication of errors detected by the channel-specific decoder to the individual audio and video decoders may be performed in various ways and such communication paths are not shown in the diagram. The prototypical decoder design does not imply any normative requirement for the design of a transport stream decoder. Indeed non-audio/video data is also allowed, but not shown.

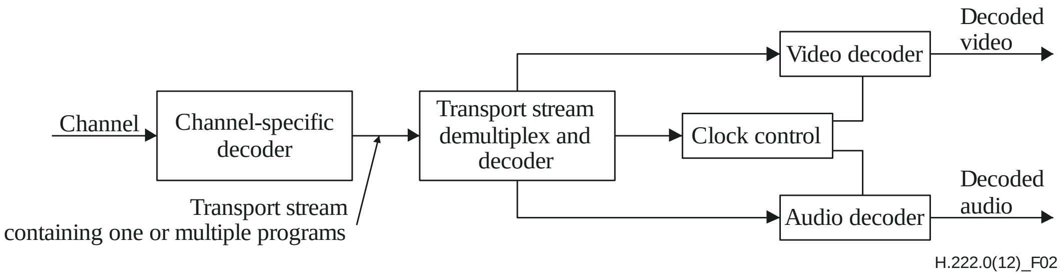


Figure Intro. 2 – Prototypical transport demultiplexing and decoding example

Figure Intro. 3 illustrates the second case, where a transport stream containing multiple programs is converted into a transport stream containing a single program. In this case the re-multiplexing operation may necessitate the correction of program clock reference (PCR) values to account for changes in the PCR locations in the bit stream.

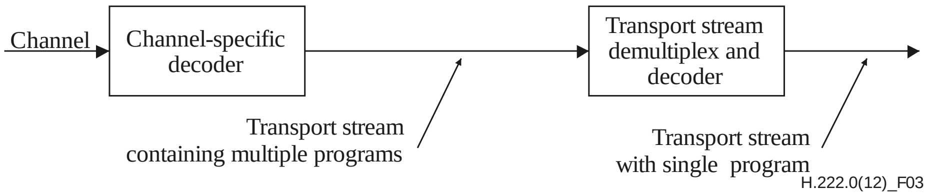


Figure Intro. 3 – Prototypical transport multiplexing example

Figure Intro. 4 illustrates a case in which a multi-program transport stream is first demultiplexed and then converted into a program stream.

Figures Intro. 3 and Intro. 4 indicate that it is possible and reasonable to convert between different types and configurations of transport streams. There are specific fields defined in the transport stream and program stream syntax which facilitate the conversions illustrated. There is no requirement that specific implementations of demultiplexors or decoders include all of these functions.

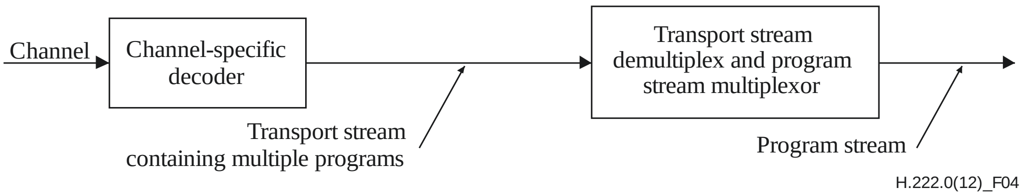


Figure Intro. 4 – Prototypical transport stream to program stream conversion

Intro. 2 Program stream

The program stream is a stream definition which is tailored for communicating or storing one program of coded data and other data in environments where errors are very unlikely, and where processing of system coding, e.g., by software, is a major consideration.

Program streams may be either fixed or variable rate. In either case, the constituent elementary streams may be either fixed or variable rate. The syntax and semantics constraints on the stream are identical in each case. The program stream rate is defined by the values and locations of the system clock reference (SCR) and mux_rate fields.

A prototypical audio/video program stream decoder system is depicted in Figure Intro. 5. The architecture is not unique – system decoder functions including decoder timing control might as equally well be distributed among elementary stream decoders and the channel-specific decoder – but this figure is useful for discussion. The prototypical decoder design does not imply any normative requirement for the design of a program stream decoder. Indeed non-audio/video data is also allowed, but not shown.

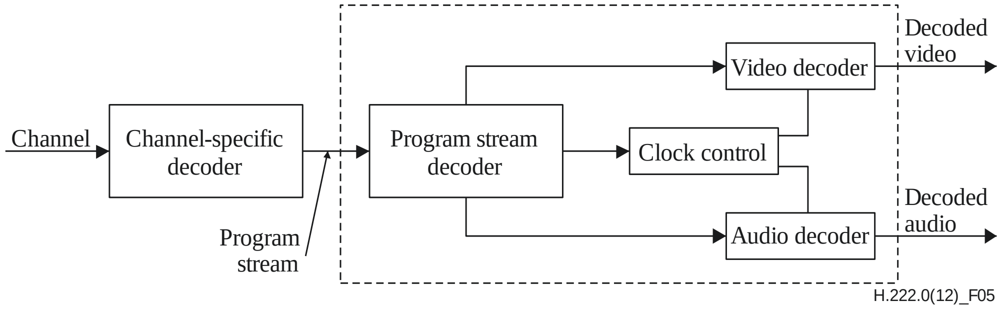


Figure Intro. 5 – Prototypical decoder for program streams

The prototypical decoder for program streams shown in Figure Intro. 5 is composed of system, video and audio decoders conforming to Parts 1, 2 and 3, respectively, of ISO/IEC 13818. In this decoder, the multiplexed coded representation of one or more audio and/or video streams is assumed to be stored or communicated on some channel in some channel-specific format. The channel-specific format is not governed by this Recommendation | International Standard, nor is the channel-specific decoding part of the prototypical decoder.

The prototypical decoder accepts as input a program stream and relies on a program stream decoder to extract timing information from the stream. The program stream decoder demultiplexes the stream, and the elementary streams so produced serve as inputs to video and audio decoders, whose outputs are decoded video and audio signals. Included in the design, but not shown in the figure, is the flow of timing information among the program stream decoder, the video and audio decoders, and the channel-specific decoder. The video and audio decoders are synchronized with each other and with the channel using this timing information.

Program streams are constructed in two layers: a system layer and a compression layer. The input stream to the program stream decoder has a system layer wrapped about a compression layer. Input streams to the video and audio decoders have only the compression layer.

Operations performed by the prototypical decoder either apply to the entire program stream ("multiplex-wide operations"), or to individual elementary streams ("stream-specific operations"). The program stream system layer is divided into two sub-layers, one for multiplex-wide operations (the pack layer), and one for stream-specific operations (the PES packet layer).

Intro. 3 Conversion between transport stream and program stream

It may be possible and reasonable to convert between transport streams and program streams by means of PES packets. This results from the specification of transport stream and program stream as embodied in 2.4.1 and 2.5.1 of the normative requirements of this Recommendation | International Standard. PES packets may, with some constraints, be mapped directly from the payload of one multiplexed bit stream into the payload of another multiplexed bit stream. It is possible to identify the correct order of PES packets in a program to assist with this if the program_packet_sequence_counter is present in all PES packets.

Certain other information necessary for conversion, e.g., the relationship between elementary streams, is available in tables and headers in both streams. Such data, if available, shall be correct in any stream before and after conversion.

Intro. 4 Packetized elementary stream

Transport streams and program streams are each logically constructed from PES packets, as indicated in the syntax definitions in 2.4.3.6. PES packets shall be used to convert between transport streams and program streams; in some cases the PES packets need not be modified when performing such conversions. PES packets may be much larger than the size of a transport stream packet.

A continuous sequence of PES packets of one elementary stream with one stream ID may be used to construct a PES Stream. When PES packets are used to form a PES stream, they shall include elementary stream clock reference (ESCR) fields and elementary stream rate (ES_Rate) fields, with constraints as defined in 2.4.3.8. The PES stream data shall be contiguous bytes from the elementary stream in their original order. PES streams do not contain some necessary system information which is contained in program streams and transport streams. Examples include the information in the pack header, system header, program stream map, program stream directory, program map table, and elements of the transport stream packet syntax.

The PES stream is a logical construct that may be useful within implementations of this Recommendation | International Standard; however, it is not defined as a stream for interchange and interoperability. Applications requiring streams containing only one elementary stream can use program streams or transport streams which each contain only one elementary stream. These streams contain all of the necessary system information. Multiple program streams or transport streams, each containing a single elementary stream, can be constructed with a common time base and therefore carry a complete program, i.e., with audio and video.

Intro. 5 Timing model

Systems, video and audio all have a timing model in which the end-to-end delay from the signal input to an encoder to the signal output from a decoder is a constant. This delay is the sum of encoding, encoder buffering, multiplexing, communication or storage, demultiplexing, decoder buffering, decoding, and presentation delays. As part of this timing model all video pictures and audio samples are presented exactly once, unless specifically coded to the contrary, and the inter-picture interval and audio sample rate are the same at the decoder as at the encoder. The system stream coding contains timing information which can be used to implement systems which embody constant end-to-end delay. It is possible to implement decoders which do not follow this model exactly; however, in such cases it is the decoder's responsibility to perform in an acceptable manner. The timing is embodied in the normative specifications of this Recommendation | International Standard, which must be adhered to by all valid bit streams, regardless of the means of creating them.

All timing is defined in terms of a common system clock, referred to as a system time clock (STC). In the program stream this clock may have an exactly specified ratio to the video or audio sample clocks, or it may have an operating frequency which differs slightly from the exact ratio while still providing precise end-to-end timing and clock recovery.

In the transport stream the system clock frequency is constrained to have the exactly specified ratio to the audio and video sample clocks at all times; the effect of this constraint is to simplify sample rate recovery in decoders.

Intro. 6 Conditional access

Encryption and scrambling for conditional access to programs encoded in the program and transport streams is supported by the system data stream definitions. Conditional access mechanisms are not specified here. The stream definitions are designed so that implementation of practical conditional access systems is reasonable, and there are some syntactical elements specified which provide specific support for such systems.

Intro. 7 Multiplex-wide operations

Multiplex-wide operations include the coordination of data retrieval of the channel, the adjustment of clocks, and the management of buffers. The tasks are intimately related. If the rate of data delivery of the channel is controllable, then data delivery may be adjusted so that decoder buffers neither overflow nor underflow; but if the data rate is not controllable, then elementary stream decoders must slave their timing to the data received from the channel to avoid overflow or underflow.

Program streams are composed of packs whose headers facilitate the above tasks. Pack headers specify intended times at which each byte is to enter the program stream Decoder from the channel, and this target arrival schedule serves as a reference for clock correction and buffer management. The schedule need not be followed exactly by decoders, but they must compensate for deviations about it.

Similarly, transport streams are composed of transport stream packets with headers containing information which specifies the times at which each byte is intended to enter a transport stream decoder from the channel. This schedule provides exactly the same function as that which is specified in the program stream.

An additional multiplex-wide operation is a decoder's ability to establish what resources are required to decode a transport stream or program stream. The first pack of each program stream conveys parameters to assist decoders in this task. Included, for example, are the stream's maximum data rate and the highest number of simultaneous video channels. The transport stream likewise contains globally useful information.

The transport stream and program stream each contain information which identifies the pertinent characteristics of, and relationships between, the elementary streams which constitute each program. Such information may include the language spoken in audio channels, as well as the relationship between video streams when multi-layer video coding is implemented.

Intro. 8 Individual stream operations (PES packet layer)

The principal stream-specific operations are:

1) demultiplexing; and
2) synchronizing playback of multiple elementary streams.

Intro. 8.1 Demultiplexing

On encoding, program streams are formed by multiplexing elementary streams, and transport streams are formed by multiplexing elementary streams, program streams, or the contents of other transport streams. Elementary streams may include private, reserved, and padding streams in addition to audio and video streams. The streams are temporally subdivided into packets, and the packets are serialized. A PES packet contains coded bytes from one and only one elementary stream.

In the program stream both fixed and variable packet lengths are allowed subject to constraints as specified in 2.5.1 and 2.5.2. For transport streams the packet length is 188 bytes. Both fixed and variable PES packet lengths are allowed, and will be relatively long in most applications.

On decoding, demultiplexing is required to reconstitute elementary streams from the multiplexed program stream or transport stream. Stream_id codes in program stream packet headers, and packet ID codes in the transport stream make this possible.

Intro. 8.2 Synchronization

Synchronization among multiple elementary streams is accomplished with presentation time stamps (PTSs) in the program stream and transport streams. Time stamps are generally in units of 90 kHz, but the system clock reference (SCR), the program clock reference (PCR) and the optional elementary stream clock reference (ESCR) have extensions with a resolution of 27 MHz. Decoding of N-elementary streams is synchronized by adjusting the decoding of streams to a common master time base rather than by adjusting the decoding of one stream to match that of another. The master time base may be one of the N-decoders' clocks, the data source's clock, or it may be some external clock.

Each program in a transport stream, which may contain multiple programs, may have its own time base. The time bases of different programs within a transport stream may be different.

Because PTSs apply to the decoding of individual elementary streams, they reside in the PES packet layer of both the transport streams and program streams. End-to-end synchronization occurs when encoders save time stamps at capture time, when the time stamps propagate with associated coded data to decoders, and when decoders use those time stamps to schedule presentations.

Synchronization of a decoding system with a channel is achieved through the use of the SCR in the program stream and by its analogue, the PCR, in the transport stream. The SCR and PCR are time stamps encoding the timing of the bit stream itself, and are derived from the same time base used for the audio and video PTS values from the same program. Since each program may have its own time base, there are separate PCR fields for each program in a transport stream containing multiple programs. In some cases it may be possible for programs to share PCR fields. Refer to 2.4.4, program-specific information (PSI), for the method of identifying which PCR is associated with a program. A program shall have one and only one PCR time base associated with it.

Intro. 8.3 Relation to compression layer

The PES packet layer is independent of the compression layer in some senses, but not in all. It is independent in the sense that PES packet payloads need not start at compression layer start codes, as defined in Parts 2 and 3 of ISO/IEC 13818. For example, video start codes may occur anywhere within the payload of a PES packet, and start codes may be split by a PES packet header. However, time stamps encoded in PES packet headers apply to presentation times of compression layer constructs (namely, presentation units). In addition, when the elementary stream data conforms to Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 13818-3, the PES_packet_data_bytes shall be byte aligned to the bytes of this Recommendation | International Standard.

Intro. 9 System reference decoder

Part 1 of ISO/IEC 13818 employs a "system target decoder" (STD), one for transport streams (refer to 2.4.2) referred to as "transport system target decoder" (T-STD) and one for program streams (refer to 2.5.2) referred to as "program system target decoder" (P-STD), to provide a formalism for timing and buffering relationships. Because the STD is parameterized in terms of Rec. ITU-T H.222.0 | ISO/IEC 13818-1 fields (for example, buffer sizes) each elementary stream leads to its own parameterization of the STD. Encoders shall produce bit streams that meet the appropriate STD's constraints. Physical decoders may assume that a stream plays properly on its STD. The physical decoder must compensate for ways in which its design differs from that of the STD.

Intro. 10 Applications

The streams defined in this Recommendation | International Standard are intended to be as useful as possible to a wide variety of applications. Application developers should select the most appropriate stream.

Modern data communications networks may be capable of supporting Rec. ITU-T H.222.0 | ISO/IEC 13818-1 video and ISO/IEC 13818 audio. A real-time transport protocol is required. The program stream may be suitable for transmission on such networks.

The program stream is also suitable for multimedia applications on CD-ROM. Software processing of the program stream may be appropriate.

The transport stream may be more suitable for error-prone environments, such as those used for distributing compressed bit-streams over long-distance networks and in broadcast systems.

Many applications require storage and retrieval of Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstreams on various digital storage media (DSM). A digital storage media command and control (DSM-CC) protocol is specified in Annex B and Part 6 of ISO/IEC 13818 in order to facilitate the control of such media.

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – Generic coding of moving pictures and associated audio information: Systems

SECTION 1 – GENERAL

1.1 Scope

This Recommendation | International Standard specifies the system layer of the coding. It was developed principally to support the combination of the video and audio coding methods defined in Parts 2 and 3 of ISO/IEC 13818. The system layer supports six basic functions:

1) the synchronization of multiple compressed streams on decoding;
2) the interleaving of multiple compressed streams into a single stream;
3) the initialization of buffering for decoding start up;
4) continuous buffer management;
5) time identification;
6) multiplexing and signalling of various components in a system stream.

A Rec. ITU-T H.222.0 | ISO/IEC 13818-1 multiplexed bit stream is either a transport stream or a program stream. Both streams are constructed from PES packets and packets containing other necessary information. Both stream types support multiplexing of video and audio compressed streams from one program with a common time base. The transport stream additionally supports the multiplexing of video and audio compressed streams from multiple programs with independent time bases. For almost error-free environments the program stream is generally more appropriate, supporting software processing of program information. The transport stream is more suitable for use in environments where errors are likely.

A Rec. ITU-T H.222.0 | ISO/IEC 13818-1 multiplexed bit stream, whether a transport stream or a program stream, is constructed in two layers: the outermost layer is the system layer, and the innermost is the compression layer. The system layer provides the functions necessary for using one or more compressed data streams in a system. The video and audio parts of this Specification define the compression coding layer for audio and video data. Coding of other types of data is not defined by this Specification, but is supported by the system layer provided that the other types of data adhere to the constraints defined in 2.7.

1.2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

1.2.1 Identical Recommendations | International Standards

– Recommendation ITU-T H.262 (2000) | ISO/IEC 13818-2:2000, *Information technology – Generic coding of moving pictures and associated audio information: Video*.

1.2.2 Paired Recommendations | International Standards equivalent in technical content

– Recommendation ITU-T H.264 (2014), *Advanced video coding for generic audiovisual services*.

ISO/IEC 14496-10:2014, *Information technology – Coding of audio-visual objects – Part 10: Advanced video coding*.

– Recommendation ITU-T H.265 (2015), *High efficiency video coding.*

ISO/IEC 23008-2:2015, *Information technology – High efficiency coding and media delivery in heterogeneous environments – Part 2: High efficiency video coding*.

– Recommendation ITU-T T.171 (1996), *Protocols for interactive audiovisual services: coded representation of multimedia and hypermedia objects*.

ISO/IEC 13522-1:1997, *Information technology – Coding of Multimedia and Hypermedia information – Part 1: MHEG object representation – Base notation (ASN.1)*.

1.2.3 Additional references

– Recommendation ITU-T J.17 (1988), *Pre-emphasis used on sound-programme circuits*.

– Recommendation ITU-T T.800 (2002) | ISO/IEC 15444-1:2004, *Information technology – JPEG 2000 image coding system: Core coding system.*

– Recommendation ITU-R BT.470-7 (2005), *Conventional analogue television systems*.

– Recommendation ITU-R BT.601-6 (2007), *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16.9 aspect ratios*.

– Recommendation ITU-R BT.709-6 (2015), *Parameter values for the HDTV standards for production and international programme exchange*.

– Recommendation ITU-R BT.1886 (2011), *Reference electro-optical transfer function for flat panel displays used in HDTV studio production.*

– Recommendation ITU-R BT.2100 (2016), *Image parameter values for high dynamic range television for use in production and international programme exchange*.

– ISO 639-2:1998, *Codes for the representation of names of languages – Part 2: Alpha-3 code*.

– ISO 8859-1:1998, *Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1*.

– ISO 15706:2002, *Information and documentation – International Standard Audiovisual Number (ISAN)*.

– ISO/IEC 11172-1:1993, *Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 1: Systems*.

– ISO/IEC 11172-2:1993, *Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 2: Video*.

– ISO/IEC 11172-3:1993, *Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 3: Audio*.

– ISO/IEC 13818-3:1998, *Information technology – Generic coding of moving pictures and associated audio information – Part 3: Audio*.

– ISO/IEC 13818-6:1998, *Information technology – Generic coding of moving pictures and associated audio information – Part 6: Extensions for DSM-CC*.

– ISO/IEC 13818-7:2006, *Information technology – Generic coding of moving pictures and associated audio information – Part 7: Advanced Audio Coding (AAC)*.

– ISO/IEC 13818-11:2004, *Information technology – Generic coding of moving pictures and associated audio information – Part 11: IPMP on MPEG-2 systems*.

– ISO/IEC 14496-1:2010, *Information technology – Coding of audio-visual objects – Part 1: Systems*.

– ISO/IEC 14496-2:2004, *Information technology – Coding of audio-visual objects – Part 2: Visual*.

– ISO/IEC 14496-3:2009, *Information technology – Coding of audio-visual objects – Part 3: Audio*.

– ISO/IEC 14496-17:2006, *Information technology, Coding of audio-visual objects – Part 17: Streaming text format*.

– ISO/IEC 23001-8:2016, *Information technology – MPEG systems technologies – Part 8: Coding-independent code-points.*

– ISO/IEC 23001-10:2015, *Information technology – MPEG systems technologies – Part 10: Carriage of timed metadata metrics of media in ISO base media file format.*

– ISO/IEC 23001-11:2015, *Information technology – MPEG systems technologies – Part 11: Energy-efficient media consumption (Green Metadata)*.

– ISO/IEC 23003-3:2012, *Information technology – MPEG audio technologies – Part 3: Unified speech and audio coding.*

– ISO/IEC 23003-4:2015, *Information technology – MPEG audio technologies – Part 4: Dynamic Range Control*.

– ISO/IEC 23008-3:2015, *Information technology – High efficiency coding and media delivery in heterogeneous environments – Part 3: 3D audio.*

- ISO/IEC 23009-1:2014 – *Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats*.
- ISO/PRF 15706-2:2007, *Information and documentation – International Standard Audiovisual Number (ISAN) – Part 2: Version identifier*.
- IEC Publication 60908:1999, *Audio recording – Compact disc digital audio system*.
- IETF RFC 3986 (2005), *Uniform Resource Identifier (URI): Generic Syntax*.
- IETF RFC 5484 (2009), *Associating Time-Codes with RTP Streams*.

SECTION 2 – TECHNICAL ELEMENTS

2.1 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply. If specific to a Part, this is parenthetically noted.

2.1.1 access unit (system): A coded representation of a presentation unit. In the case of audio, an access unit is the coded representation of an audio frame, whereby each audio frame carries data from one or more audio channels; an audio frame may carry for example one mono channel, or two stereo channels or seven surround sound channels.

In the case of video, an access unit includes all the coded data for a picture, and any stuffing that follows it, up to but not including the start of the next access unit. If a picture is not preceded by a group_start_code or a sequence_header_code, the access unit begins with the picture start code. If a picture is preceded by a group_start_code and/or a sequence_header_code, the access unit begins with the first byte of the first of these start codes. If it is the last picture preceding a sequence_end_code in the bitstream, all bytes between the last byte of the coded picture and the sequence_end_code (including the sequence_end_code) belong to the access unit.

For the definition of an access unit for Rec. ITU-T H.264 | ISO/IEC 14496-10 video, see the AVC access unit definition in 2.1.3.

In the case of an ISO/IEC 14496-17 text stream, see ISO/IEC 14496-17 for the definition of an access unit.

2.1.2 AVC 24-hour picture (system): An advanced video coding (AVC) access unit with a presentation time that is more than 24 hours in the future. For the purpose of this definition, AVC access unit n has a presentation time that is more than 24 hours in the future if the difference between the initial arrival time $t_{ai}(n)$ and the DPB output time $t_{o,dpb}(n)$ is more than 24 hours.

2.1.3 AVC access unit (system): An access unit as defined for byte streams in Rec. ITU-T H.264 | ISO/IEC 14496-10 with the constraints specified in 2.14.1.

2.1.4 AVC slice (system): A byte_stream_nal_unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 with nal_unit_type values of 1 or 5, or a byte_stream_nal_unit data structure with nal_unit_type value of 2 and any associated byte_stream_nal_unit data structures with nal_unit_type equal to 3 and/or 4.

2.1.5 AVC still picture (system): An AVC still picture consists of an AVC access unit containing an IDR picture, preceded by SPS and PPS NAL units that carry sufficient information to correctly decode the IDR picture. Preceding an AVC still picture, there shall be another AVC still picture or an end of sequence NAL unit terminating a preceding coded video sequence unless the AVC still picture is the very first access unit in the video stream.

2.1.6 AVC video sequence (system): Coded video sequence as defined in 3.30 of Rec. ITU-T H.264 | ISO/IEC 14496-10.

2.1.7 AVC video stream (system): A Rec. ITU-T H.264 | ISO/IEC 14496-10 stream. An AVC video stream consists of one or more AVC video sequences. An AVC video stream may also result from re-assembling video sub-bitstreams.

2.1.8 AVC video sub-bitstream of MVC: The video sub-bitstream that contains the base view as defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, containing all VCL NAL units associated with the minimum value of view order index present in each AVC video sequence of the AVC video stream. The AVC video sub-bitstream of MVC may additionally contain the associated NAL units with nal_unit_type syntax element equal to 14 (prefix NAL units), as defined for MVC in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10.

2.1.9 AVC video sub-bitstream of MVCD: The video sub-bitstream that contains the base view as defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, containing all VCL NAL units associated with the minimum value of view order index present in each AVC video sequence of the AVC video stream. The AVC video sub-bitstream of MVCD may additionally contain the associated NAL units with nal_unit_type syntax element equal to 14 (prefix NAL units), as defined for MVC in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10.

2.1.10 AVC video sub-bitstream of SVC: The video sub-bitstream that contains the base layer as defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10 and that shall additionally contain NAL units with nal_unit_type equal to 14 (prefix NAL units)) as defined for scalable video coding (SVC) in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10. The AVC video sub-bitstream of SVC contains all VCL NAL units associated with dependency_id equal to 0.

2.1.11 bitrate: The rate at which the compressed bit stream is delivered from the channel to the input of a decoder.

2.1.12 byte aligned: A bit in a coded bit stream is byte-aligned if its position is a multiple of 8-bits from the first bit in the stream.

2.1.13 channel: A digital medium that stores or transports a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream.

2.1.14 coded B-frame: A B-frame picture or a pair of B-field pictures.

2.1.15 coded frame: A coded frame is a coded I-frame, coded B-frame or a coded P-frame.

2.1.16 coded I-frame: An I-frame picture or a pair of field pictures where the first field picture is an I-picture and the second field picture is either an I-picture or a P-picture.

2.1.17 coded P-frame: A P-frame picture or a pair of P-field pictures.

2.1.18 coded representation: A data element as represented in its encoded form.

2.1.19 compression: Reduction in the number of bits used to represent an item of data.

2.1.20 constant bitrate: Operation where the bitrate is constant from start to finish of the compressed bit stream.

2.1.21 constrained system parameter stream (CSPS) (system): A program stream for which the constraints defined in 2.7.9 apply.

2.1.22 cyclic redundancy check (CRC): The CRC to verify the correctness of data.

2.1.23 data element: An item of data as represented before encoding and after decoding.

2.1.24 decoded stream: The decoded reconstruction of a compressed bit stream.

2.1.25 decoder: An embodiment of a decoding process.

2.1.26 decoding (process): The process defined in this Recommendation | International Standard that reads an input-coded bit stream and outputs decoded pictures or audio samples.

2.1.27 decoding time-stamp (DTS) (system): A field that may be present in a PES packet header that indicates the time that an access unit is decoded in the system target decoder.

2.1.28 digital storage media (DSM): A digital storage or transmission device or system.

2.1.29 DSM-CC: Digital storage media command and control.

2.1.30 entitlement control message (ECM): Entitlement control messages are private conditional access information which specify control words and possibly other, typically stream-specific, scrambling and/or control parameters.

2.1.31 entitlement management message (EMM): Entitlement management messages are private conditional access information which specify the authorization levels or the services of specific decoders. They may be addressed to single decoders or groups of decoders.

2.1.32 editing: The process by which one or more compressed bit streams are manipulated to produce a new compressed bit stream. Edited bit streams meet the same requirements as streams which are not edited.

2.1.33 elementary stream (ES) (system): A generic term for one of the coded video, coded audio or other coded bit streams in PES packets. One elementary stream is carried in a sequence of PES packets with one and only one stream_id.

2.1.34 elementary stream clock reference (ESCR) (system): A time stamp in the PES stream from which decoders of PES streams may derive timing.

2.1.35 encoder: An embodiment of an encoding process.

2.1.36 encoding (process): A process, not specified in this Recommendation | International Standard, that reads a stream of input pictures or audio samples and produces a coded bit stream conforming to this Recommendation.

2.1.37 entropy coding: Variable length lossless coding of the digital representation of a signal to reduce redundancy.

2.1.38 event: An event is defined as a collection of elementary streams with a common time base, an associated start time, and an associated end time.

2.1.39 fast forward playback (video): The process of displaying a sequence, or parts of a sequence, of pictures in display-order faster than real-time.

2.1.40 forbidden: The term "forbidden", when used in the clauses of this Recommendation | International Standard defining the coded bit stream, indicates that the value specified shall never be used.

2.1.41 green access unit – An access unit that contains dynamic metadata as defined in 6.2.1 of ISO/IEC 23001-11.

2.1.42 HEVC 24-hour picture (system): An *HEVC access unit* with a presentation time that is more than 24 hours in the future. For the purpose of this definition, *HEVC access unit* n has a presentation time that is more than 24 hours in the future if the difference between the initial arrival time $t_{ai}(n)$ and the DPB output time $t_{o,dpb}(n)$ is more than 24 hours.

2.1.43 HEVC access unit: An access unit as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2 with the constraints specified in 2.17.1.

2.1.44 HEVC base layer: HEVC layer with nuh_layer_id equal to 0.

2.1.45 HEVC base sub-partition: HEVC video sub-bitstream that is also a conforming bitstream as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2, which contains all VCL NAL units and the associated non-VCL NAL units of an HEVC base layer up to a target highest TemporalId identified by a target HEVC operation point.

2.1.46 HEVC complete temporal representation: A sub-layer representation as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 that contains all temporal sub-layers up to the temporal sub-layer with TemporalId equal to sps_max_sub_layers_minus1+1 as included in the active sequence parameter set, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.47 HEVC dependent slice segment: An *HEVC slice segment* with the syntax element dependent_slice_segment_flag in the slice header set to a value equal to 1, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.48 HEVC enhancement sub-partition: One HEVC layer with a particular value of nuh_layer_id greater than 0 in the NAL unit header syntax element or an HEVC temporal video sub-bitstream or HEVC temporal video subset thereof, of which the HEVC layer aggregation with an HEVC base sub-partition and zero or more other HEVC sub-partitions, according to HEVC layer list, results in a valid HEVC layered video stream.

2.1.49 HEVC highest temporal sub-layer representation: The sub-layer representation of the temporal sub-layer with the highest value of TemporalId, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2, in the associated *HEVC temporal video sub-bitstream* or *HEVC temporal video subset*.

2.1.50 HEVC independent slice segment: An *HEVC slice segment* with the syntax element dependent_slice_segment_flag in the slice header set to a value 0 or inferred to be equal to 0, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.51 HEVC layer: HEVC video sub-bitstream that contains all VCL NAL units with a particular value of nuh_layer_id in the NAL unit header syntax element and associated non-VCL NAL units, as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.52 HEVC layer aggregation: Successive HEVC layer component aggregation of all HEVC layer components in an HEVC video sequence.

2.1.53 HEVC layer component: VCL NAL units and the associated non-VCL NAL units of an HEVC access unit which belong to an HEVC sub-partition.

2.1.54 HEVC layer component aggregation: Concatenation of all HEVC layer components with the same output time from all HEVC sub-partitions indicated in an HEVC layer list in the order indicated by the HEVC layer list, resulting in a valid HEVC access unit as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.55 HEVC layer list: Ordered list of HEVC sub-partitions for a target HEVC operation point of which the HEVC layer aggregation results in a valid HEVC layered video stream.

NOTE – An HEVC layer list is signalled for each target HEVC operation point using the HEVC operation point descriptor.

2.1.56 HEVC layered video stream: HEVC video stream that contains all VCL NAL units and associated non-VCL NAL units conforming to one or more profiles defined in Annex G or Annex H of Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.57 HEVC operation point: Operation point based on a target highest TemporalId, and a target layer identifier list as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2.

NOTE – Rec. ITU-T H.265 | ISO/IEC 23008-2 specifies the sub-bitstream extraction process for an operation point according to which the operation point is a conforming bitstream. An operation point is associated with an HEVC layered video stream or HEVC base layer.

2.1.58 HEVC slice: An *HEVC independent slice segment* and zero or more subsequent *HEVC dependent slice segments* preceding the next *HEVC independent slice segment* (if any) within the same *HEVC access unit*.

2.1.59 HEVC slice segment: A byte_stream_nal_unit with nal_unit_type in the range of 0 to 9 and 16 to 23, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.60 HEVC still picture (system): An HEVC still picture consists of an *HEVC access unit* containing an IDR picture preceded by VPS, SPS and PPS NAL units, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2, that carry sufficient information to correctly decode this IDR picture. Preceding an HEVC still picture, there shall be another HEVC still picture or an end of sequence NAL unit terminating a preceding coded video sequence, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.61 HEVC sub-partition: Either an HEVC base sub-partition or an HEVC enhancement sub-partition.

NOTE – An HEVC sub-partition can either be an HEVC temporal video sub-bitstream if it includes VCL NAL units with the minimum value of TemporalId (i.e., including TemporalId equal to 0), or it can be an HEVC temporal video subset, if it complements an HEVC base sub-partition or HEVC enhancement sub-partition with the same target layer identifier.

2.1.62 HEVC temporal enhancement sub-partition: An HEVC temporal video subset of the same HEVC layer as another HEVC enhancement sub-partition of the same HEVC video stream which contains one or more complementary temporal sub-layers, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.63 HEVC temporal video sub-bitstream: An HEVC video sub-bitstream that contains all VCL NAL units and associated non-VCL NAL units of the temporal sub-layer of the same layer, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2, associated with TemporalId equal to 0 and which may additionally contain all VCL NAL units and associated non-VCL NAL units of all temporal sub-layers of the same layer associated with a contiguous range of TemporalId from 1 to a value equal to or smaller than sps_max_sub_layers_minus1 included in the active sequence parameter set, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.64 HEVC temporal video subset: An HEVC video sub-bitstream that contains all VCL NAL units and the associated non-VCL NAL units of one or more temporal sub-layers of the same layer, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2, with each temporal sub-layer not being present in the corresponding HEVC temporal video sub-bitstream and TemporalId associated with each temporal sub-layer forming a contiguous range of values that is equal to or smaller than sps_max_sub_layers_minus1 included in the active sequence parameter set, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2..

NOTE – According to the constraints for the transport of HEVC specified in 2.17.1, each temporal sub-layer of an *HEVC video stream* is present either in the HEVC temporal video sub-bitstream or in exactly one HEVC temporal video subset which is carried in a set of elementary streams that are associated by hierarchy descriptors or HEVC hierarchy extension descriptors. This prevents multiple inclusions of the same temporal sub-layer and allows aggregation of the HEVC temporal video sub-bitstream with associated HEVC temporal video subsets according to the hierarchy descriptors, as specified in 2.17.3 and according to the hierarchy descriptors or HEVC hierarchy extension descriptors, as specified in 2.17.4.

2.1.65 HEVC tile of slices: One or more consecutive *HEVC slices* which form the coded representation of a tile, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.66 HEVC video sequence (system): A coded video sequence as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2.

2.1.67 HEVC video stream: Byte stream as specified in Rec. ITU-T H. 265 | ISO/IEC 23008-2 Annex B.

NOTE – This term represents either a byte stream as specified in Annex B of the first version of Rec. ITU-T H.265 | ISO/IEC 23008-2 or an HEVC layered video sub-bitstream.

2.1.68 HEVC video sub-bitstream: A subset of the NAL units of an HEVC video stream in their original order.

2.1.69 JPEG 2000 (J2K) video access unit: An access unit defined in Rec. ITU-T T.800 (2002)/Amd.3 (2010) | ISO/IEC 15444-1:2004/Amd.3:2010 which includes all the parameters required to decode the access unit and display the decoded data.

2.1.70 J2K still picture (system): J2K video access unit as defined in 2.1.69 with constraints as specified in S.2.

2.1.71 J2K video elementary stream: Video elementary stream consisting of a succession of J2K video access units.

2.1.72 J2K video sequence: J2K video elementary stream where all the access units have the same profile/level, J2K video access unit coding parameters and video parameters.

2.1.73 layer (video and systems): One of the levels in the data hierarchy of the video and system specifications defined in Parts 1 and 2 of this Recommendation | International Standard.

2.1.74 metadata: Information to describe audiovisual content and data essence in a format defined by ISO or any other authority.

2.1.75 metadata access unit: A global structure within metadata that defines the fraction of metadata that is intended to be decoded at a specific instant in time. The internal structure of a metadata Access Unit is defined by the format of the metadata.

2.1.76 metadata application format: Identifies the format of the application that uses the metadata; signals application specific information for transport of metadata.

2.1.77 metadata decoder configuration information: Data needed by a receiver to decode a specific metadata service. Depending on the format of the metadata, decoder configuration information may or may not be needed.

2.1.78 metadata format: Identifies the coding format of metadata.

2.1.79 metadata service: A coherent set of metadata of the same format delivered to a receiver for a specific purpose.

2.1.80 metadata service id: Identifier of a specific metadata service; used for some transport methods of the metadata.

2.1.81 metadata stream: The concatenation or collection of metadata Access Units from one or more metadata services.

2.1.82 (multiplexed) stream (system): A bit stream composed of 0 or more elementary streams combined in a manner that conforms to this Recommendation | International Standard.

2.1.83 MVC base view sub-bitstream: The MVC base view sub-bitstream is defined to contain the AVC video sub-bitstream of MVC conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10 and one additional MVC video sub-bitstream associated with an MVC view_id subset including the view order index that immediately follows the view order index associated with the base view.

NOTE – The MVC base view sub-bitstream is also an AVC video stream where no re-assembly is required before decoding.

2.1.84 MVC operation point: An MVC operation point is identified by a temporal_id value representing a target temporal level and a set of view_id values representing the target output views. One MVC operation point is associated with an AVC video stream which conforms to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10. The AVC video stream associated with an MVC operation point is re-assembled from a set consisting of one or more of the following items: AVC video sub-bitstream of MVC, MVC base view sub-bitstream, MVC video sub-bitstreams.

2.1.85 MVC slice (system): A byte_stream_nal_unit with nal_unit_type syntax element equal to 20 of an AVC video stream which conforms to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE – As specified in Rec. ITU-T H.264 | ISO/IEC 14496-10, the value of svc_extension_flag is set equal to 0 for coded video sequences conforming to one or more profiles specified in Annex H. MVC slices should not include NAL units for which nal_unit_type is equal to 20 with svc_extension_flag equal to 1.

2.1.86 MVC video sub-bitstream: The MVC video sub-bitstream is defined to be all VCL NAL units with nal_unit_type equal to 20 associated with the same multiview video coding (MVC) view_id subset of an advanced video coding (AVC) video stream and associated non-VCL NAL units which conform to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE – In contrast to a sub-bitstream as specified in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, an MVC video sub-bitstream according to this Specification is not necessarily a decodable MVC video sub-bitstream. The one exception is when an MVC video sub-bitstream is also an MVC base view sub-bitstream. Re-assembling MVC video sub-bitstreams in an increasing order of view order index, starting from the lowest value of view order index up to any value of view order index, results in a decodable AVC video stream.

2.1.87 MVC view_id subset: A set of one or more view_id values, as defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10 in the NAL unit header syntax element, associated with one set of consecutive view order index values.

NOTE – An MVC video sub-bitstream or MVC base view sub-bitstream based on a specific MVC view_id subset may not include view components for all view_id values included in that MVC view_id subset. One or more view order index values may be skipped if the view associated with a missing view order index value is not required for decoding the transmitted views.

2.1.88 MVCD base view sub-bitstream: The MVCD base view sub-bitstream is defined to contain the AVC video sub-bitstream of MVCD conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10 and one additional MVCD video sub-bitstream associated with an MVCD view_id subset including the view order index that immediately follows the view order index associated with the base view.

NOTE – The MVCD base view sub-bitstream is also an AVC video stream where no re-assembly is required before decoding.

2.1.89 MVCD slice (system): A byte_stream_nal_unit with nal_unit_type syntax element equal to 21 of an AVC video stream which conforms to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10.

2.1.90 MVCD video sub-bitstream: The MVCD video sub-bitstream is defined to be all VCL NAL units with nal_unit_type equal to 21 associated with the same MVCD view_id subset of an AVC video stream and associated non-VCL NAL units which conform to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE – In contrast to a sub-bitstream as specified in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, an MVCD video sub-bitstream according to this Specification is not necessarily a decodable MVCD video sub-bitstream. The one exception is when an MVCD video sub-bitstream is also an MVCD base view sub-bitstream. Re-assembling MVCD video sub-bitstreams in an increasing order of view order index, starting from the lowest value of view order index up to any value of view order index, results in a decodable AVC video stream.

2.1.91 MVC view-component subset: The VCL NAL units of an AVC access unit associated with the same MVC view_id subset and associated non-VCL NAL units.

NOTE – Re-assembling MVC view-component subsets ordered according to the view order index, starting from the minimum view order index up to the highest view order index present in the access unit, while reordering the non-VCL NAL units conforming to the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10, results in an AVC access unit.

2.1.92 MVCD view-component subset: The VCL NAL units of an AVC access unit associated with the same MVCD view_id subset and associated non-VCL NAL units.

NOTE – Re-assembling MVCD view-component subsets ordered according to the view order index, starting from the minimum view order index up to the highest view order index present in the access unit, while reordering the non-VCL NAL units conforming to the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10, results in an AVC access unit.

2.1.93 MVCD view_id subset: A set of one or more view_id values, as defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10 in the NAL unit header syntax element, associated with one set of consecutive view order index values.

NOTE – An MVCD video sub-bitstream or MVCD base view sub-bitstream based on a specific MVCD view_id subset may not include view components for all view_id values included in that MVCD view_id subset. One or more view order index values may be skipped if the view associated with a missing view order index value is not required for decoding the transmitted views.

2.1.94 pack (system): A pack consists of a pack header followed by zero or more packets. It is a layer in the system coding syntax described in 2.5.3.3.

2.1.95 packet data (system): Contiguous bytes of data from an elementary stream present in a packet.

2.1.96 packet identifier (PID) (system): A unique integer value used to identify elementary streams of a program in a single or multi-program transport stream as described in 2.4.3.

2.1.97 padding (audio): A method to adjust the average length of an audio frame in time to the duration of the corresponding PCM samples, by conditionally adding a slot to the audio frame.

2.1.98 payload: Payload refers to the bytes which follow the header bytes in a packet. For example, the payload of some transport stream packets includes a PES_packet_header and its PES_packet_data_bytes, or pointer_field and PSI sections, or private data; but a PES_packet_payload consists of only PES_packet_data_bytes. The transport stream packet header and adaptation fields are not payload.

2.1.99 PES (system): An abbreviation for a Packetized Elementary Stream.

2.1.100 PES packet (system): The data structure used to carry elementary stream data. A PES packet consists of a PES packet header followed by a number of contiguous bytes from an elementary data stream. It is a layer in the system coding syntax described in 2.4.3.6.

2.1.101 PES packet header (system): The leading fields in a PES packet up to and not including the PES_packet_data_byte fields, where the stream is not a padding stream. In the case of a padding stream the PES packet header is similarly defined as the leading fields in a PES packet up to and not including padding_byte fields.

2.1.102 packetized elementary stream (PES) (system): A PES system consists of PES packets, all of whose payloads consist of data from a single elementary stream, and all of which have the same stream_id. Specific semantic constraints apply. Refer to Intro. 4.

2.1.103 presentation time-stamp (PTS) (system): A field that may be present in a PES packet header that indicates the time that a presentation unit is presented in the system target decoder.

2.1.104 presentation unit (PU) (system): A decoded audio access unit or a decoded picture.

2.1.105 program (system): A program is a collection of program elements. Program elements may be elementary streams. Program elements need not have any defined time base; those that do, have a common time base and are intended for synchronized presentation.

2.1.106 program clock reference (PCR) (system): A time stamp in the transport stream from which decoder timing is derived.

2.1.107 program element (system): A generic term for one of the elementary streams or other data streams that may be included in a program.

2.1.108 program-specific information (PSI) (system): PSI consists of normative data which is necessary for the demultiplexing of transport streams and the successful regeneration of programs and is described in 2.4.4. An example of privately defined PSI data is the non-mandatory network information table.

2.1.109 quality access unit – An access unit that contains dynamic quality metadata as defined in ISO/IEC 23001-10.

2.1.110 random access: The process of beginning to read and decode the coded bit stream at an arbitrary point.

2.1.111 reserved: The term "reserved", when used in the clauses defining the coded bit stream, indicates that the value may be used in the future for ISO defined extensions. Unless otherwise specified within this Recommendation | International Standard, all reserved bits shall be set to '1'.

2.1.112 scrambling (system): The alteration of the characteristics of a video, audio or coded data stream in order to prevent unauthorized reception of the information in a clear form. This alteration is a specified process under the control of a conditional access system.

2.1.113 service-compatible: This is defined as 'simulcast' of two stereoscopic views which do not include scalable or temporal coding. The two views are independently compressed using MPEG-2 video or AVC or both and can be decoded independently.

2.1.114 source stream: A single non-multiplexed stream of samples before compression coding.

2.1.115 splicing (system): The concatenation, performed on the system level, of two different elementary streams. The resulting system stream conforms totally to this Recommendation | International Standard. The splice may result in discontinuities in timebase, continuity counter, PSI, and decoding.

2.1.116 start codes (system): 32-bit codes embedded in the coded bit stream. They are used for several purposes including identifying some of the layers in the coding syntax. Start codes consist of a 24-bit prefix (0x000001) and an 8-bit stream_id as shown in Table 2-22.

2.1.117 STD input buffer (system): A first-in first-out buffer at the input of a system target decoder for storage of compressed data from elementary streams before decoding.

2.1.118 still picture: A still picture consists of a video sequence, coded as defined in Rec. ITU-T H.262 | ISO/IEC 13818-2, ISO/IEC 11172-2 or ISO/IEC 14496-2, that contains exactly one coded picture which is intra-coded. This picture has an associated PTS and in case of coding according to ISO/IEC 11172-2, Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 14496-2, the presentation time of succeeding pictures, if any, is later than that of the still picture by at least two picture periods.

2.1.119 SVC dependency representation: The VCL NAL units of an AVC access unit associated with the same value of dependency_id which is provided as part of the NAL unit header or the associated prefix NAL unit header, and the associated non-VCL NAL units. Re-assembling SVC dependency representations in a consecutive order of dependency_id starting from the lowest value of dependency_id present in the access unit up to any value of dependency_id present in the access unit, while reordering the non-VCL NAL units conforming to the order of NAL units within an access unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10, results in an AVC access unit.

2.1.120 SVC slice (system): A byte_stream_nal_unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 with nal_unit_type equal to 20 of an AVC video stream which conforms to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE – As specified in Rec. ITU-T H.264 | ISO/IEC 14496-10, the value of svc_extension_flag is set equal to 1 for coded video sequences conforming to one or more profiles specified in Annex G. SVC slices should not include NAL units for which nal_unit_type is equal to 20 with svc_extension_flag equal to 0.

2.1.121 SVC video sub-bitstream: The video sub-bitstream that contains VCL NAL units with nal_unit_type equal to 20 with the same NAL unit header syntax element dependency_id not equal to 0.

2.1.122 system header (system): The system header is a data structure defined in 2.5.3.5 that carries information summarizing the system characteristics of Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program stream.

2.1.123 system clock reference (SCR) (system): A time stamp in the program stream from which decoder timing is derived.

2.1.124 system target decoder (STD) (system): A hypothetical reference model of a decoding process used to define the semantics of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 multiplexed bit stream.

2.1.125 time-stamp (system): A term that indicates the time of a specific action such as the arrival of a byte or the presentation of a Presentation Unit.

2.1.126 transport stream packet header (system): The leading fields in a transport stream packet, up to and including the continuity_counter field.

2.1.127 variable bitrate: An attribute of transport streams or program streams wherein the rate of arrival of bytes at the input to a decoder varies with time.

2.1.128 video sub-bitstream: A video sub-bitstream is defined to be all VCL NAL units associated with the same value of dependency_id of an AVC video stream which conforms to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10 and all associated non-VCL NAL units in decoding order as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10. Re-assembling video sub-bitstreams in a consecutive order of dependency_id, starting from the dependency_id equal to 0 up to any value of dependency_id, results in an AVC video stream. A video sub-bitstream shall have the AVC byte stream format as defined in Annex B of Rec. ITU-T H.264 | ISO/IEC 14496-10.

2.1.129 view order index: An index that indicates the decoding order of MVC view components in an AVC access unit as defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10 or MVCD view components in an AVC access unit as defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10. The association of view order index values to the NAL unit header syntax element view_id is indicated for an AVC video sequence in the sequence parameter set MVC extension as defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10 or in the sequence parameter set MVCD extension as defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10.

2.2 Symbols and abbreviations

The mathematical operators used to describe this Recommendation | International Standard are similar to those used in the C-programming language. However, integer division with truncation and rounding are specifically defined. The bitwise operators are defined assuming two's-complement representation of integers. Numbering and counting loops generally begin from 0.

2.2.1 Arithmetic operators

+	Addition
–	Subtraction (as a binary operator) or negation (as a unary operator)
++	Increment
– –	Decrement
* or ×	Multiplication
^	Power
/	Integer division with truncation of the result toward 0. For example, 7/4 and –7/–4 are truncated to 1 and –7/4 and 7/–4 are truncated to –1.
//	Integer division with rounding to the nearest integer. Half-integer values are rounded away from 0 unless otherwise specified. For example 3//2 is rounded to 2, and –3//2 is rounded to –2.
DIV	Integer division with truncation of the result towards – ∞.
%	Modulus operator. Defined only for positive numbers.
Sign( )	Sign(x) = 1 x > 0 0 x = = 0 –1 x < 0
NINT( )	Nearest integer operator. Returns the nearest integer value to the real-valued argument. Half-integer values are rounded away from 0.
sin	Sine
cos	Cosine
exp	Exponential
√	Square root
$\log_{10}$	Logarithm to base ten
$\log_e$	Logarithm to base e

2.2.2 Assignment

=	Assignment operator

2.2.3 Bitwise operators

&	AND
|	OR

>> Shift right with sign extension

<< Shift left with 0 fill

2.2.4 Constants

π 3.14159265359

e 2.71828182845

2.2.5 Logical operators

|| Logical OR

&& Logical AND

! Logical NOT

2.2.6 Mnemonics

The following mnemonics are defined to describe the different data types used in the coded bit-stream.

bslbf Bit string, left bit first, where "left" is the order in which bit strings are written in this Recommendation | International Standard. Bit strings are written as a string of 1s and 0s within single quote marks, e.g., '1000 0001'. Blanks within a bit string are for ease of reading and have no significance.

NOTE – In some tables of this specification, two bit strings are used to define a range of values. In other tables, bit strings are used to specify integer values. In these cases, each bit string is equivalent to a binary number of which the least significant bit (unity position) equals the rightmost bit of the bit string and the significance of bits increases from right to left.

ch Channel

gr Granule of 3 * 32 sub-band samples in audio Layer II, 18 * 32 sub-band samples in audio Layer III.

main_data The main_data portion of the bit stream contains the scale factors, Huffman encoded data, and ancillary information.

main_data_beg This gives the location in the bit stream of the beginning of the main_data for the frame. The location is equal to the ending location of the previous frame's main_data plus 1 bit. It is calculated from the main_data_end value of the previous frame.

part2_length This value contains the number of main_data bits used for scale factors.

rpchof Remainder polynomial coefficients, highest order first

sb Sub-band

scfsi Scalefactor selector information

switch_point_l Number of scalefactor band (long block scalefactor band) from which point on window switching is used

switch_point_s Number of scalefactor band (short block scalefactor band) from which point on window switching is used

tcimsbf Two's complement integer, msb (sign) bit first

uimsbf Unsigned integer, most significant bit first

vlclbf Variable length code, left bit first, where "left" refers to the order in which the variable length codes are written

window Number of actual time slot in case of block_type = = 2, 0 ≤ window ≤ 2.

The byte order of multi-byte words is most significant byte first.

2.2.7 Range operator

.. Range operator. n .. m defines the inclusive range of numbers from n up to m

2.2.8 Relational operators

> Greater than

≥ Greater than or equal to

< Less than

≤ Less than or equal to

= =	Equal to
!=	Not equal to
max [,...,]	The maximum value in the argument list
min [,...,]	The minimum value in the argument list

2.3 Method of describing bit stream syntax

The bit streams retrieved by the decoder are described in 2.4.1 and 2.5.1. Each data item in the bit stream is in bold type. It is described by its name, its length in bits, and a mnemonic for its type and order of transmission.

The action caused by a decoded data element in a bit stream depends on the value of that data element and on data elements previously decoded. The decoding of the data elements and definition of the state variables used in their decoding are described in the clauses containing the semantic description of the syntax. The following constructs are used to express the conditions when data elements are present, and are in normal type.

Note this syntax uses the "C"-code convention that a variable or expression evaluating to a non-zero value is equivalent to a condition that is true:

while (condition) {
  data_element
    . . .
}
	If the condition is true, then the group of data elements occurs next in the data stream. This repeats until the condition is not true.

do {
  data_element
    . . .
}
while (condition)
	The data element always occurs at least once. The data element is repeated until the condition is not true.

if (condition) {
  data_element
    . . .
}
	If the condition is true, then the first group of data elements occurs next in the data stream.

else {
  data_element
    . . .
}
	If the condition is not true, then the second group of data elements occurs next in the data stream.

for (i = 0; i < n; i++) {
  data_element
    . . .
}
	The group of data elements occurs n times. Conditional constructs within the group of data elements may depend on the value of the loop control variable i, which is set to zero for the first occurrence, incremented to 1 for the second occurrence, and so forth.

As noted, the group of data elements may contain nested conditional constructs. For compactness, the {} are omitted when only one data element follows:

data_element []	data_element [] is an array of data. The number of data elements is indicated by the context.
data_element [n]	data_element [n] is the n+1th element of an array of data.
data_element [m][n]	data_element [m][n] is the m+1,n+1th element of a two-dimensional array of data.
data_element [l][m][n]	data_element [l][m][n] is the l+1,m+1,n+1th element of a three-dimensional array of data.
data_element [m..n]	is the inclusive range of bits between bit m and bit n in the data_element.

While the syntax is expressed in procedural terms, it should not be assumed that either Figure 2-1 or Figure 2-2 implements a satisfactory decoding procedure. In particular, they define a correct and error-free input bitstream. Actual decoders must include a means to look for start codes and sync bytes (transport stream) in order to begin decoding correctly, and to identify errors, erasures or insertions while decoding. The methods to identify these situations, and the actions to be taken, are not standardized.

2.4 Transport stream bitstream requirements

2.4.1 Transport stream coding structure and parameters

The Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport stream coding layer allows one or more programs to be combined into a single stream. Data from each elementary stream are multiplexed together with information that allows synchronized presentation of the elementary streams within a program.

A transport stream consists of one or more programs. Audio and video elementary streams consist of access units.

Elementary stream data is carried in PES packets. A PES packet consists of a PES packet header followed by packet data. PES packets are inserted into transport stream packets. The first byte of each PES packet header is located at the first available payload location of a transport stream packet.

The PES packet header begins with a 32-bit start-code that also identifies the stream or stream type to which the packet data belongs. The PES packet header may contain decoding and presentation time stamps (DTS and PTS). The PES packet header also contains other optional fields. The PES packet data field contains a variable number of contiguous bytes from one elementary stream.

Transport stream packets begin with a 4-byte prefix, which contains a 13-bit packet ID (PID), defined in Table 2-2. The PID identifies, via the program-specific information (PSI) tables, the contents of the data contained in the transport stream packet. Transport stream packets of one PID value carry data of one and only one elementary stream.

The PSI tables are carried in the transport stream. There are six PSI tables:

- Program association table (PAT);
- Program map table (MPT);
- Conditional access table (CAT);
- Network information table (NIT);
- Transport stream description table (TSDT);
- IPMP control information table.

These tables contain the necessary and sufficient information to demultiplex and present programs. The program map table in Table 2-33 specifies, among other information, which PIDs, and therefore which elementary streams, are associated to form each program. This table also indicates the PID of the transport stream packets which carry the PCR for each program. The conditional access table shall be present if scrambling is employed. The network information table is optional and its contents are not specified by this Recommendation | International Standard. The IPMP control information table shall be present if IPMP as described in ISO/IEC 13818-11 is used by any of the components in the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream.

Transport stream packets may be null packets. Null packets are intended for padding of transport streams. They may be inserted or deleted by re-multiplexing processes and, therefore, the delivery of the payload of null packets to the decoder cannot be assumed.

This Recommendation | International Standard does not specify the coded data which may be used as part of conditional access systems. This Specification does, however, provide mechanisms for program service providers to transport and identify this data for decoder processing, and to reference correctly data which are specified by this Specification. This type of support is provided both through transport stream packet structures and in the conditional access table (refer to Table 2-32).

2.4.2 Transport stream system target decoder

2.4.2.1 General

The semantics of the transport stream specified in 2.4.3 and the constraints on these semantics specified in 2.7 require exact definitions of byte arrival and decoding events and the times at which these occur. The definitions needed are set out in this Recommendation | International Standard using a hypothetical decoder known as the transport stream system target decoder (T-STD). Informative Annex D contains further explanation of the T-STD.

The T-STD is a conceptual model used to define these terms precisely and to model the decoding process during the construction or verification of transport streams. The T-STD is defined only for this purpose. There are three types of decoders in the T-STD: video, audio, and systems. Figure 2-1 illustrates an example. Neither the architecture of the T-STD nor the timing described precludes uninterrupted, synchronized play-back of transport streams from a variety of decoders with different architectures or timing schedules.

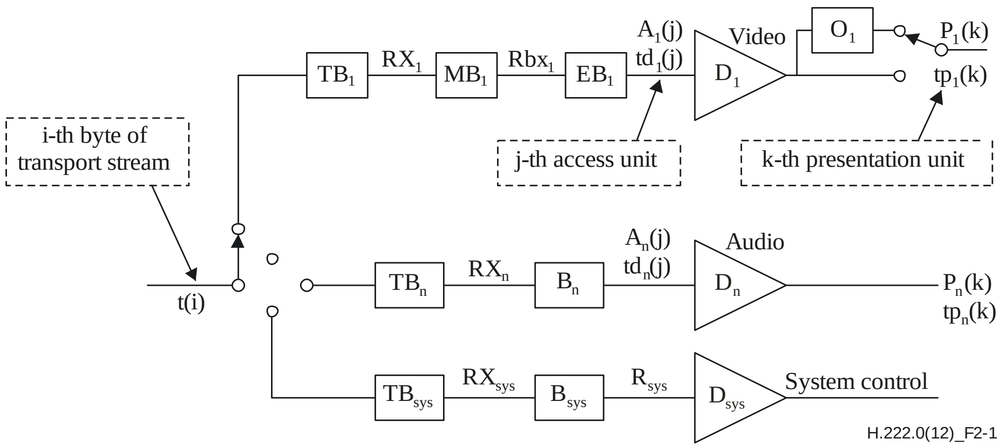


Figure 2-1 – Transport stream system target decoder notation

The following notation is used to describe the transport stream system target decoder and is partially illustrated in Figure 2-1 above.

i, i′, i″	are indices to bytes in the transport stream. The first byte has index 0.
j	is an index to access units in the elementary streams.
k, k′, k″	are indices to presentation units in the elementary streams.
n	is an index to the elementary streams.
p	is an index to transport stream packets in the transport stream.
t(i)	indicates the time in seconds at which the i-th byte of the transport stream enters the system target decoder. The value t(0) is an arbitrary constant.
PCR(i)	is the time encoded in the PCR field measured in units of the period of the 27-MHz system clock where i is the byte index of the final byte of the program_clock_reference_base field.
$A_n(j)$	is the j-th access unit in elementary stream n. $A_n(j)$ is indexed in decoding order.
$td_n(j)$	is the decoding time, measured in seconds, in the system target decoder of the j-th access unit in elementary stream n.
$P_n(k)$	is the k-th presentation unit in elementary stream n. $P_n(k)$ results from decoding $A_n(j)$. $P_n(k)$ is indexed in presentation order.
$tp_n(k)$	is the presentation time, measured in seconds, in the system target decoder of the k-th presentation unit in elementary stream n.
t	is time measured in seconds.
$F_n(t)$	is the fullness, measured in bytes, of the system target decoder input buffer for elementary stream n at time t.
B_n	is the main buffer for elementary stream n. It is present only for audio elementary streams.
BS_n	is the size of buffer, B_n, measured in bytes.
B_{sys}	is the main buffer in the system target decoder for system information for the program that is in the process of being decoded.
BS_{sys}	is the size of B_{sys}, measured in bytes.
MB_n	is the multiplexing buffer, for elementary stream n. It is present only for video elementary streams.
MBS_n	is the size of MB_n, measured in bytes.
EB_n	is the elementary stream buffer for elementary stream n. It is present only for video elementary streams.
EBS_n	is the size of the elementary stream buffer EB_n, measured in bytes.
TB_{sys}	is the transport buffer for system information for the program that is in the process of being decoded.
TBS_{sys}	is the size of TB_{sys}, measured in bytes.
TB_n	is the transport buffer for elementary stream n.

TBS_n is the size of TB_n, measured in bytes.

D_{sys} is the decoder for system information in program stream n.

D_n is the decoder for elementary stream n.

O_n is the re-order buffer for video elementary stream n.

R_{sys} is the rate at which data are removed from B_{sys}.

Rx_n is the rate at which data are removed from TB_n.

Rbx_n is the rate at which PES packet payload data are removed from MB_n when the leak method is used. Defined only for video elementary streams.

$Rbx_n(j)$ is the rate at which PES packet payload data are removed from MB_n when the vbv_delay method is used. Defined only for video elementary streams.

Rx_{sys} is the rate at which data are removed from TB_{sys}.

R_{es} is the video elementary stream rate coded in a sequence header.

2.4.2.2 System clock frequency

Timing information referenced in the T-STD is carried by several data fields defined in this Specification. Refer to 2.4.3.4 and 2.4.3.6. In PCR fields this information is coded as the sampled value of a program's system clock. The PCR fields are carried in the adaptation field of the transport stream packets with a PID value equal to the PCR_PID defined in the TS_program_map_section of the program being decoded.

Practical decoders may reconstruct this clock from these values and their respective arrival times. The following are minimum constraints which apply to the program's system clock frequency as represented by the values of the PCR fields when they are received by a decoder.

The value of the system clock frequency is measured in Hz and shall meet the following constraints:

$$27\,000\,000 - 810 \le \text{system\_clock\_frequency} \le 27\,000\,000 + 810$$

$$\text{rate of change of system\_clock\_frequency with time} \le 75 \times 10^{-3} \text{ Hz/s}$$

NOTE – Sources of coded data should follow a tighter tolerance in order to facilitate compliant operation of consumer recorders and playback equipment.

A program's system_clock_frequency may be more accurate than required. Such improved accuracy may be transmitted to the decoder via the system clock descriptor described in 2.6.20.

Bit rates defined in this Specification are measured in terms of system_clock_frequency. For example, a bit rate of 27 000 000 bits per second in the T-STD would indicate that one byte of data is transferred every eight (8) cycles of the system clock.

The notation "system_clock_frequency" is used in several places in this Specification to refer to the frequency of a clock meeting these requirements. For notational convenience, equations in which PCR, PTS, or DTS appear, lead to values of time which are accurate to some integral multiple of (300×2^{33}/system_clock_frequency) seconds. This is due to the encoding of PCR timing information as 33 bits of 1/300 of the system clock frequency plus 9 bits for the remainder, and encoding as 33 bits of the system clock frequency divided by 300 for PTS and DTS.

2.4.2.3 Input to the transport stream system target decoder

Input to the transport stream system target decoder (T-STD) is a transport stream. A transport stream may contain multiple programs with independent time bases. However, the T-STD decodes only one program at a time. In the T-STD model all timing indications refer to the time base of that program.

Data from the transport stream enters the T-STD at a piecewise constant rate. The time t(i) at which the i-th byte enters the T-STD is defined by decoding the program clock reference (PCR) fields in the input stream, encoded in the transport stream packet adaptation field of the program to be decoded and by counting the bytes in the complete transport stream between successive PCRs of that program. The PCR field (see equation 2-1) is encoded in two parts: one, in units of the period of 1/300 times the system clock frequency, called program_clock_reference_base (see equation 2-2), and one in units of the system clock frequency called program_clock_reference_extension (see equation 2-3). The values encoded in these are computed by PCR_base(i) (see equation 2-2) and PCR_ext(i) (see equation 2-3) respectively. The value encoded in the PCR field indicates the time t(i), where i is the index of the byte containing the last bit of the program_clock_reference_base field.

Specifically:

$$PCR(i) = PCR\_base(i) \times 300 + PCR\_ext(i) \quad (2\text{-}1)$$

where:

$$PCR\_base(i) = ((system\_clock\_frequency \times t(i)) DIV\ 300) \% 2^{33} \quad (2\text{-}2)$$

$$PCR\_ext(i) = ((system\_clock\_frequency \times t(i)) DIV\ 1) \% 300 \quad (2\text{-}3)$$

For all other bytes the input arrival time, t(i) shown in equation 2-4 below, is computed from PCR(i″) and the transport rate at which data arrive, where the transport rate is determined as the number of bytes in the transport stream between the bytes containing the last bit of two successive program_clock_reference_base fields of the same program divided by the difference between the time values encoded in these same two PCR fields.

$$t(i) = \frac{PCR(i'')}{system\_clock\_frequency} + \frac{i - i''}{transport\_rate(i)} \quad (2\text{-}4)$$

where:

i is the index of any byte in the transport stream for i″ < i < i′.

i″ is the index of the byte containing the last bit of the most recent program_clock_reference_base field applicable to the program being decoded.

PCR(i″) is the time encoded in the program clock reference base and extension fields in units of the system clock.

The transport rate for any byte i between byte i″ and byte i′ is given by:

$$transport\_rate(i) = \frac{((i' - i'') \times system\_clock\_frequency)}{PCR(i') - PCR(i'')} \quad (2\text{-}5)$$

where:

i′ is the index of the byte containing the last bit of the immediately following program_clock_reference_base field applicable to the program being decoded.

In the case of a timebase discontinuity, indicated by the discontinuity_indicator in the transport packet adaptation field, the definition given in equation 2-4 and equation 2-5 for the time of arrival of bytes at the input to the T-STD is not applicable between the last PCR of the old timebase and the first PCR of the new timebase. In this case the time of arrival of these bytes is determined according to equation 2-4 with the modification that the transport rate used is that applicable between the last and next to last PCR of the old timebase.

A tolerance is specified for the PCR values. The PCR tolerance is defined as the maximum inaccuracy allowed in received PCRs. This inaccuracy may be due to imprecision in the PCR values or to PCR modification during re-multiplexing. It does not include errors in packet arrival time due to network jitter or other causes. The PCR tolerance is ± 500 ns.

In the T-STD model, the inaccuracy will be reflected as an inaccuracy in the calculated transport rate using equation 2-5.

Transport streams with multiple programs and variable rate

Transport streams may contain multiple programs which have independent time bases. Separate sets of PCRs, as indicated by the respective PCR_PID values, are required for each such independent program, and therefore the PCRs cannot be co-located. The transport stream rate is piecewise constant for the program entering the T-STD. Therefore, if the transport stream rate is variable it can only vary at the PCRs of the program under consideration. Since the PCRs, and therefore the points in the transport Stream where the rate varies, are not co-located, the rate at which the transport stream enters the T-STD would have to differ depending on which program is entering the T-STD. Therefore, it is not possible to construct a consistent T-STD delivery schedule for an entire transport stream when that transport stream contains multiple programs with independent time bases and the rate of the transport stream is variable. It is straightforward, however, to construct constant bit rate transport streams with multiple variable rate programs.

2.4.2.4 Buffering

Complete transport stream packets containing system information, for the program selected for decoding, enter the system transport buffer, TB_{sys}, at the transport stream rate. These include transport stream packets whose PID values are 0, 1, 2 or 3, and all transport stream packets identified via the program association table (see Table 2-30) as having the program_map_PID value for the selected program. Network information table (NIT) data as specified by the NIT PID is not transferred to TB_{sys}.

NOTE 1 – Size of IPMP control information table could be large, and the repetition rate of this table should be adjusted to meet the buffer requirement.

All bytes that enter the buffer TB_n are removed at the rate Rx_n specified below. Bytes which are part of the PES packet header or its contents are delivered to the main buffer B_n for audio elementary streams and system data, and to the multiplexing buffer MB_n for video elementary streams. Other bytes are not, and may be used to control the system. Duplicate transport stream packets are not delivered to B_n, MB_n, or B_{sys}.

The buffer TB_n is emptied as follows:

– When there is no data in TB_n, Rx_n is equal to zero.

– Otherwise for video:

$$Rx_n = 1,2 \times R_{\max}[profile, level]$$

where:

R_{max}[profile, level] is specified according to the profile and level which can be found in Table 8-13 of Rec. ITU-T H.262 | ISO/IEC 13818-2. This table specifies the upper bound of the rate of each elementary video stream within a specific profile and level.

Rx_n is equal to 1, 2 × R_{max} for ISO/IEC 11172-2 constrained parameter video streams, where R_{max} refers to the maximum bitrate for a constrained parameters bitstream in ISO/IEC 11172-2.

For ISO/IEC 13818-7 ADTS audio:

Number of Channels	Rx_n [bit/s]
1 .. 2	2 000 000
3 .. 8	5 529 600
9 .. 12	8 294 400
13 .. 48	33 177 600

Channels: The number of full-bandwidth audio output channels plus the number of independently switched coupling channel elements within the same elementary audio stream. For example, in the typical case that there are no independently switched coupling channel elements, mono is 1 channel, stereo is 2 channels and 5.1 channel surround is five channels (the low-frequency effects (LFE) channel is not counted).

For other audio,

$$Rx_n = 2 \times 10^6 \text{ bits per second}$$

For systems data:

$$Rx_n = 1 \times 10^6 \text{ bits per second}$$

Rx_n is measured with respect to the system clock frequency.

Complete transport stream packets containing system information, for the program selected for decoding, enter the system transport buffer, TB_{sys}, at the transport stream rate. These include transport stream packets whose PID values are 0, 1, 2 and 3 (if present), and all transport stream packets identified via the program association table (PAT) (see Table 2-30) as having the program_map_PID value for the selected program. Network information table (NIT) data as specified by the NIT PID is not transferred to TB_{sys}.

Bytes are removed from TB_{sys} at the rate Rx_{sys} and delivered to B_{sys}. Each byte is transferred instantaneously.

Duplicate transport stream packets are not delivered to B_{sys}.

Transport packets which do not enter any TB_n or TB_{sys} are discarded.

The transport buffer size is fixed at 512 bytes.

The elementary stream buffer sizes EBS_1 through EBS_n are defined for video as equal to the vbv_buffer_size as it is carried in the sequence header. Refer to the summary of constrained parameters in ISO/IEC 11172-2 and Table 8-14 of Rec. ITU-T H.262 | ISO/IEC 13818-2.

NOTE 2 – In the following equations, unit conversion should be implicitly performed as appropriate. Values expressed in bits are implicitly converted into values expressed in bytes by: *number_of_bytes* = (*number_of_bits* +7) / 8.

The multiplexing buffer size MBS_1 through MBS_n are defined for video as follows:

For Low and Main level:

$$MBS_n = BS_{mux} + BS_{oh} + VBV_{\max}[profile,\ level] - vbv\_buffer\_size$$

where BS_{oh}, PES packet overhead buffering is defined as:

$$BS_{oh} = (1/750)\,\text{seconds} \times R_{\max}[profile,\ level]$$

and BS_{mux}, additional multiplex buffering is defined as:

$$BS_{mux} = 0.004\ \text{seconds} \times R_{\max}[profile,\ level]$$

and where VBV_{max}[profile, level] is defined in Table 8-14 of Rec. ITU-T H.262 | ISO/IEC 13818-2 and R_{max}[profile, level] is defined in Table 8-13 of Rec. ITU-T H.262 | ISO/IEC 13818-2, and vbv buffer size is carried in the sequence header described in 6.2.2 of Rec. ITU-T H.262 | ISO/IEC 13818-2.

For High 1440 and High level:

$$MBS_n = BS_{mux} + BS_{oh}$$

where BS_{oh} is defined as:

$$BS_{oh} = (1/750)\ \text{seconds} \times R_{\max}[profile, level]$$

and BS_{mux} is defined as:

$$BS_{mux} = 0.004\text{seconds} \times R_{\max}[profile, level]$$

and where R_{max}[profile, level] is defined in Table 8-13 of Rec. ITU-T H.262 | ISO/IEC 13818-2.

For Constrained Parameters ISO/IEC 11172-2 bitstreams:

$$MBS_n = BS_{mux} + BS_{oh} + vbv\_\max - vbv\_buffer\_size$$

where BS_{oh} is defined as:

$$BS_{oh} = (1/750)\text{seconds} \times R_{\max}$$

and BS_{mux} is defined as:

$$BS_{mux} = 0.004\text{seconds} \times R_{\max}$$

and where R_{max} and vbv_max refer to the maximum bitrate and the maximum vbv_buffer_size for a Constrained Parameters bitstream in ISO/IEC 11172-2 respectively.

A portion BS_{mux} = 4 ms × R_{max}[profile, level] of the MBS_n is allocated for buffering to allow multiplexing. The remainder is available for BS_{oh} and may also be available for initial multiplexing.

NOTE 3 – Buffer occupancy by PES packet overhead is directly bounded in PES streams by the PES-STD which is defined in 2.5.2.4. It is possible, but not necessary, to utilize PES streams to construct transport streams.

Buffer BS_n

The main buffer sizes BS_1 through BS_n are defined as follows.

Audio

For ISO/IEC 13818-7 ADTS audio:

Number of Channels	BS_n [bytes]
1 .. 2	3 584
3 .. 8	8 976
9 .. 12	12 804
13 .. 48	51 216

Channels: The number of full-bandwidth audio output channels plus the number of independently switched coupling channel elements within the same elementary audio stream. For example, in the typical case that there are no independently switched coupling channel elements, mono is 1 channel, stereo is 2 channels and 5.1 channel surround is 5 channels (the LFE channel is not counted).

For other audio:

$$BS_n = BS_{mux} + BS_{dec} + BS_{oh} = 3584 \ bytes$$

The size of the access unit decoding buffer BS_{dec}, and the PES packet overhead buffer BS_{oh} are constrained by:

$$BS_{dec} + BS_{oh} \leq 2848 \ bytes$$

A portion (736 bytes) of the 3584 byte buffer is allocated for buffering to allow multiplexing. The rest, 2848 bytes, are shared for access unit buffering BS_{dec}, BS_{oh} and additional multiplexing.

Systems

The main buffer B_{sys} for system data is of size BS_{sys} = 1536 bytes.

Video

For video elementary streams, data is transferred from MB_n to EB_n using one of two methods: the leak method or the video buffering verifier (VBV) delay method.

Leak method

The leak method transfers data from MB_n to EB_n using a leak rate R_{bx}. The leak method is used whenever any of the following is true:

- the STD descriptor (refer to 2.6.32) for the elementary stream is not present in the transport stream;
- the STD descriptor is present and the leak_valid flag has a value of '1';
- the STD descriptor is present, the leak_valid has a value of '0', and the vbv_delay fields coded in the video stream have the value 0xFFFF; or
- trick mode status is true (refer to 2.4.3.7).

For Low and Main level:

$$Rbx_n = R_{\max} \, [profile, \, level]$$

For High-1440 and High level:

$$Rbx_n = Min\{1.05 \times R_{es}, R_{\max} [profile, \, level]\}$$

For Constrained Parameters bitstream in ISO/IEC 11172-2:

$$Rbx_n = 1,\, 2 \times R_{\max}$$

where R_{max} is the maximum bit rate for a Constrained Parameters bitstream in ISO/IEC 11172-2.

If there is PES packet payload data in MB_n, and buffer EB_n is not full, the PES packet payload is transferred from MB_n to EB_n at a rate equal to Rbx_n. If EB_n is full, data are not removed from MB_n. When a byte of data is transferred from MB_n to EB_n, all PES packet header bytes that are in MB_n and immediately precede that byte, are instantaneously removed and discarded. When there is no PES packet payload data present in MB_n, no data is removed from MB_n. All data that enters MB_n leaves it. All PES packet payload data bytes enter EB_n instantaneously upon leaving MB_n.

Vbv_delay method

The vbv_delay method specifies precisely the time at which each byte of coded video data is transferred from MB_n to EB_n, using the vbv_delay values coded in the video elementary stream. The vbv_delay method is used whenever the STD descriptor (refer to 2.6.32) for this elementary stream is present in the transport stream, the leak_valid flag in the descriptor has the value '0', and vbv_delay fields coded in the video stream are not equal to 0xFFFF. If any vbv_delay values in a video sequence are not equal to 0xFFFF, none of the vbv_delay fields in that sequence shall be equal to 0xFFFF (refer to ISO/IEC 11172-2 and Rec. ITU-T H.262 | ISO/IEC 13818-2).

When the vbv_delay method is used, the final byte of the video picture start code for picture j is transferred from MB_n to the EB_n at the time $td_n(j)$ – vbv_delay(j), where $td_n(j)$ is the decoding time of picture j, as defined above, and vbv_delay(j) is the delay time, in seconds, indicated by the vbv_delay field of picture j. The transfer of bytes between the final bytes of successive picture start codes (including the final byte of the second start code), into the buffer EB_n, is at a piecewise constant rate, $R_{bx}(j)$, which is specified for each picture j. Specifically, the rate, $R_{bx}(j)$, of transfer into this buffer is given by:

$$R_{bx}(j) = NB(j)/(vbv\_delay(j) - vbv\_delay(j+1) + td_n(j+1) - td_n(j)) \quad (2\text{-}6)$$

where NB(j) is the number of bytes between the final bytes of the picture start codes (including the final byte of the second start code) of pictures j and j + 1, excluding PES packet header bytes.

NOTE 4 – vbv_delay(j + 1) and $td_n(j + 1)$ may have values that differ from those normally expected for periodic video display if the low_delay flag in the video sequence extension is set to '1'. It may not be possible to determine the correct values by examination of the bit stream.

The $R_{bx}(j)$ derived from equation 2-6 shall be less than or equal to R_{max}[profile, level] for elementary streams of stream type 0x02 (refer to Table 2-34), where R_{max}[profile, level] is defined in Rec. ITU-T H.262 | ISO/IEC 13818-2, and shall be less than or equal to the maximum bit rate allowed for constrained parameter video elementary streams of stream type 0x01, refer to ISO/IEC 11172-2.

When a byte of data is transferred from MB_n to EB_n, all PES packet header bytes that are in MB_n and immediately precede that byte are instantaneously removed and discarded. All data that enters MB_n leaves it. All PES packet payload data bytes enter EB_n instantaneously upon leaving MB_n.

Removal of access units

For each elementary stream buffer EB_n and main buffer B_n all data for the access unit that has been in the buffer longest, $A_n(j)$, and any stuffing bytes that immediately precede it that are present in the buffer at the time $td_n(j)$ are removed instantaneously at time $td_n(j)$. The decoding time $td_n(j)$ is specified in the DTS or PTS fields (refer to 2.4.3.6). Decoding times $td_n(j + 1)$, $td_n(j + 2)$, ... of access units without encoded DTS or PTS fields which directly follow access unit j may be derived from information in the elementary stream. Refer to Annex C of Rec. ITU-T H.262 | ISO/IEC 13818-2, ISO/IEC 13818-3, or ISO/IEC 11172. Also refer to 2.7.5. In the case of audio, all PES packet headers that are stored immediately before the access unit or that are embedded within the data of the access unit are removed simultaneously with the removal of the access unit. As the access unit is removed it is instantaneously decoded to a presentation unit.

System data

In the case of system data, data is removed from the main buffer B_{sys} at a rate of R_{sys} whenever there is at least 1 byte available in buffer B_{sys}.

$$R_{sys} = \max\,(80\,000 \text{ bits/s},\ transport\_rate(i) \times 8\ bits/byte/500) \quad (2\text{-}7)$$

NOTE 5 – The intention of increasing R_{sys} in the case of high transport rates is to allow an increased data rate for the Program-specific information.

Low delay

When the low_delay flag in the video sequence extension is set to '1' (see 6.2.2.3 of Rec. ITU-T H.262 | ISO/IEC 13818-2) the EB_n buffer may underflow. In this case, when the T-STD elementary stream buffer EB_n is examined at the time specified by $td_n(j)$, the complete data for the access unit may not be present in the buffer EB_n. When this case arises, the buffer shall be re-examined at intervals of two field-periods until the data for the complete access unit is present in the buffer. At this time the entire access unit shall be removed from buffer EB_n instantaneously. Overflow of buffer EB_n shall not occur.

When the low_delay_mode flag is set to '1', EB_n underflow is allowed to occur continuously without limit. The T-STD decoder shall remove access unit data from buffer EB_n at the earliest time consistent with the paragraph above and any DTS or PTS values encoded in the bit stream. Note that the decoder may be unable to re-establish correct decoding and display times as indicated by DTS and PTS until the EB_n buffer underflow situation ceases and a PTS or DTS is found in the bit stream.

Trick mode

When the DSM_trick_mode flag (2.4.3.6) is set to '1' in the PES Packet header of a packet containing the start of a B-type video access unit and the trick_mode_control field is set to '001' (slow motion) or '010' (freeze frame), or '100' (slow reverse) the B-picture access unit is not removed from the video data buffer EB_n until the last time of possibly multiple times that any field of the picture is decoded and presented. Repetition of the presentation of fields and pictures is defined in 2.4.3.8 under slow motion, slow reverse, and field_id_cntrl. The access unit is removed instantaneously from EB_n at the indicated time, which is dependent on the value of rep_cntrl.

When the DSM_trick_mode flag is set to '1' in the PES packet header of a packet containing the first byte of a picture start code, trick_mode status becomes true when that picture start code in the PES packet is removed from buffer EB_n Trick mode status remains true until a PES packet header is received by the T-STD in which the DSM_trick_mode flag is set to '0' and the first byte of the picture start code after that PES packet header is removed from buffer EB_n. When trick mode status is true, the buffer EB_n may underflow. All other constraints from normal streams are retained when trick mode status is true.

2.4.2.5 Decoding

Elementary streams buffered in B_1 through B_n and EB_1 through EB_n are decoded instantaneously by decoders D_1 through D_n and may be delayed in re-order buffers O_1 through O_n before being presented at the output of the T-STD. Re-order buffers are used only in the case of a video elementary stream when some access units are not carried in presentation order. These access units will need to be re-ordered before presentation. In particular, if $P_n(k)$ is an I-picture or a P-picture carried before one or more B-pictures, then it must be delayed in the re-order buffer, O_n, of the T-STD before being presented. Any picture previously stored in O_n is presented before the current picture can be stored. $P_n(k)$ should be delayed until the next I-picture or P-picture is decoded. While it is stored in the re-order buffer, the subsequent B-pictures are decoded and presented.

The time at which a presentation unit $P_n(k)$ is presented is $tp_n(k)$. For presentation units that do not require re-ordering delay, $tp_n(k)$ is equal to $td_n(j)$ since the access units are decoded instantaneously; this is the case, for example, for B-frames. For presentation units that are delayed, $tp_n(k)$ and $td_n(j)$ differ by the time that $P_n(k)$ is delayed in the re-order buffer, which is a multiple of the nominal picture period. Care should be taken to use adequate re-ordering delay from the beginning of video elementary streams to meet the requirements of the entire stream. For example, a stream which initially has only I- and P-pictures but later includes B-pictures should include re-ordering delay starting at the beginning of the stream.

Rec. ITU-T H.262 | ISO/IEC 13818-2 explains re-ordering of video pictures in greater detail.

2.4.2.6 Presentation

The function of a decoding system is to reconstruct presentation units from compressed data and to present them in a synchronized sequence at the correct presentation times. Although real audio and visual presentation devices generally have finite and different delays and may have additional delays imposed by post-processing or output functions, the system target decoder models these delays as zero.

In the T-STD in Figure 2-1 the display of a video presentation unit (a picture) occurs instantaneously at its presentation time, $tp_n(k)$.

In the T-STD the output of an audio presentation unit starts at its presentation time, $tp_n(k)$, when the decoder instantaneously presents the first sample. Subsequent samples in the presentation unit are presented in sequence at the audio sampling rate.

2.4.2.7 Buffer management

Transport streams shall be constructed so that conditions defined in this subclause are satisfied. This subclause makes use of the notation defined for the system target decoder.

TB_n and TB_{sys} shall not overflow. TB_n and TB_{sys} shall empty at least once every second. B_n shall not overflow nor underflow. B_{sys} shall not overflow.

EB_n shall not underflow except when the low delay flag in the video sequence extension is set to '1' (refer to 6.2.2.3 in Rec. ITU-T H.262 | ISO/IEC 13818-2) or trick_mode status is true.

When the leak method for specifying transfers is in effect, MB_n shall not overflow, and shall empty at least once every second. EB_n shall not overflow.

When the vbv_delay method for specifying transfers is in effect, MB_n shall not overflow nor underflow, and EB_n shall not overflow.

The delay of any data through the system target decoder buffers shall be less than or equal to one second except for still picture video data, ISO/IEC 14496 streams and ISO/IEC 23008-2 streams. Specifically: $td_n(j) - t(i) \leq 1$ second for all j, and all bytes i in access unit $A_n(j)$.

For still picture video data, the delay is constrained by $td_n(j) - t(i) \leq 60$ seconds for all j, and all bytes i in access unit $A_n(j)$.

For ISO/IEC 14496 and ISO/IEC 23008-2 streams, the delay is constrained by $td_n(j) - t(i) \leq 10$ seconds for all j, and all bytes i in access unit $A_n(j)$.

Definition of overflow and underflow

Let $F_n(t)$ be the instantaneous fullness of T-STD buffer B_n.

$F_n(t) = 0$ instantaneously before $t = t(0)$

Overflow does not occur if:

$$F_n(t) \leq BS_n$$

for all t and n.

Underflow does not occur if:

$$0 \leq F_n(t)$$

for all t and n.

2.4.2.8 T-STD extensions for carriage of ISO/IEC 14496 data

For decoding of ISO/IEC 14496 data carried in a transport stream the T-STD model is extended. T-STD parameters for decoding of individual ISO/IEC 14496 elementary streams are defined in 2.11.2, while 2.11.3 defines T-STD extensions and parameters for decoding of ISO/IEC 14496 scenes and associated streams.

2.4.2.9 T-STD extensions for carriage of Rec. ITU-T H.264 | ISO/IEC 14496-10 video

To define the decoding in the T-STD of Rec. ITU-T H.264 | ISO/IEC 14496-10 video streams carried in a transport stream, the T-STD model needs to be extended. The T-STD extension and T-STD parameters for decoding of AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 video streams are defined in 2.14.3.1, T-STD extension and T-STD parameters for decoding of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10 are defined in 2.14.3.5, and T-STD extension and T-STD parameters for decoding of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10 are defined in 2.14.3.7.

2.4.2.10 T-STD extensions for carriage of ISO/IEC 14496-17 text streams

To define the decoding in the T-STD of ISO/IEC 14496-17 text streams carried in a transport stream, the T-STD model needs to be extended. The T-STD extension and T-STD parameters for decoding of ISO/IEC 14496-17 text streams are defined in 2.15.3.1.

2.4.2.11 T-STD extensions for carriage of J2K video elementary streams

The interpretation, extensions, use and constraints for syntax elements in the adaptation header (2.4.3.4 and 2.4.3.5) for JPEG 2000 part 1 video are defined in S.5.

The interpretation, extensions, use and constraints for syntax elements in the PES header (2.4.3.6 and 2.4.3.7) for JPEG 2000 Part 1 video are defined in S.5.

To define the decoding of J2K video elementary streams carried in a transport stream, the T-STD model needs to be extended. The T-STD extensions and T-STD parameters for decoding of J2K video elementary streams conforming to one or more profiles defined in Rec. ITU-T T.800 (2002) | ISO/IEC 15444-1:2004 are defined in S.6.

NOTE – No extensions are specified for P-STD model, as carriage of J2K video elementary streams in program streams is not supported.

2.4.2.12 T-STD extensions for carriage of HEVC

T-STD extensions and T-STD parameters for the decoding of HEVC video streams are defined in 2.17.2 and 2.17.3. Program stream support including P-STD extensions and P-STD parameters are not specified for HEVC video streams.

2.4.2.13 T-STD extensions for carriage of MVCD video sub-bitstream

T-STD extensions and T-STD parameters for decoding of MVCD video sub-bitstreams are defined in 2.14.1 and 2.14.3.7.

NOTE – Program stream extensions are not specified for MVCD video sub-bitstreams.

2.4.2.14 T-STD extensions for carriage of MV HEVC and SHVC

T-STD extensions and T-STD parameters for decoding of HEVC layered video streams are defined in 2.17.4. Program stream support including P-STD extensions and P-STD parameters are not specified for HEVC extension video streams.

2.4.3 Specification of the transport stream syntax and semantics

The following syntax describes a stream of bytes. Transport stream packets shall be 188 bytes long.

2.4.3.1 Transport stream

See Table 2-1.

Table 2-1 – Transport stream

Syntax	No. of bits	Mnemonic
MPEG_transport_stream() {		
do {		
transport_packet()		
} while (nextbits() = = sync_byte)		
}		

2.4.3.2 Transport stream packet layer

See Table 2-2.

Table 2-2 – Transport packet of this Recommendation | International Standard

Syntax	No. of bits	Mnemonic
transport_packet(){		
sync_byte	**8**	**bslbf**
transport_error_indicator	**1**	**bslbf**
payload_unit_start_indicator	**1**	**bslbf**
transport_priority	**1**	**bslbf**
PID	**13**	**uimsbf**
transport_scrambling_control	**2**	**bslbf**
adaptation_field_control	**2**	**bslbf**
continuity_counter	**4**	**uimsbf**

Syntax	No. of bits	Mnemonic
if(adaptation_field_control = = '10' || adaptation_field_control = = '11'){		
adaptation_field()		
}		
if(adaptation_field_control = = '01' || adaptation_field_control = = '11') {		
for (i = 0; i < N; i++){		
data_byte	**8**	**bslbf**
}		
}		
}		

2.4.3.3 Semantic definition of fields in transport stream packet layer

sync_byte – The sync_byte is a fixed 8-bit field whose value is '0100 0111' (0x47). Sync_byte emulation in the choice of values for other regularly occurring fields, such as PID, should be avoided.

transport_error_indicator – The transport_error_indicator is a 1-bit flag. When set to '1' it indicates that at least 1 uncorrectable bit error exists in the associated transport stream packet. This bit may be set to '1' by entities external to the transport layer. When set to '1' this bit shall not be reset to '0' unless the bit value(s) in error have been corrected.

payload_unit_start_indicator – The payload_unit_start_indicator is a 1-bit flag which has normative meaning for transport stream packets that carry PES packets (refer to 2.4.3.6) or transport stream section data (refer to Table 2-31 in 2.4.4.4).

When the payload of the transport stream packet contains PES packet data, the payload_unit_start_indicator has the following significance: a '1' indicates that the payload of this transport stream packet will commence with the first byte of a PES packet and a '0' indicates no PES packet shall start in this transport stream packet. If the payload_unit_start_indicator is set to '1', then one and only one PES packet starts in this transport stream packet. This also applies to private streams of stream_type 6 (refer to Table 2-34).

When the payload of the transport stream packet contains transport stream section data, the payload_unit_start_indicator has the following significance: if the transport stream packet carries the first byte of a section, the payload_unit_start_indicator value shall be '1', indicating that the first byte of the payload of this transport stream packet carries the pointer_field. If the transport stream packet does not carry the first byte of a section, the payload_unit_start_indicator value shall be '0', indicating that there is no pointer_field in the payload. Refer to 2.4.4.1 and 2.4.4.2. This also applies to private streams of stream_type 5 (refer to Table 2-34).

For null packets the payload_unit_start_indicator shall be set to '0'.

The meaning of this bit for transport stream packets carrying only private data is not defined in this Specification.

transport_priority – The transport_priority is a 1-bit indicator. When set to '1' it indicates that the associated packet is of greater priority than other packets having the same PID which do not have the bit set to '1'. The transport mechanism can use this to prioritize its data within an elementary stream. Depending on the application the transport_priority field may be coded regardless of the PID or within one PID only. This field may be changed by channel-specific encoders or decoders.

PID – The PID is a 13-bit field, indicating the type of the data stored in the packet payload. PID value 0x0000 is reserved for the program association table (see Table 2-30). PID value 0x0001 is reserved for the conditional access table (see Table 2-32). PID value 0x0002 is reserved for the transport stream description table (see Table 2-36), PID value 0x0003 is reserved for IPMP control information table (see ISO/IEC 13818-11) and PID values 0x0004-0x000F are reserved. PID value 0x1FFF is reserved for null packets (see Table 2-3).

Table 2-3 – PID table

Value	Description
0x0000	Program association table
0x0001	Conditional access table
0x0002	Transport stream description table
0x0003	IPMP control information table
0x0004	Adaptive streaming information (see Note 2)
0x0005 .. 0x000F	Reserved

0x0010 .. 0x1FFE	May be assigned as network_PID, Program_map_PID, elementary_PID, or for other purposes
0x1FFF	Null packet
NOTE 1 – The transport packets with PID values 0x0000, 0x0001, and 0x0010-0x1FFE are allowed to carry a PCR. NOTE 2 – Payload syntax is defined in 5.10.3.3.5 of ISO/IEC 23009-1.	

transport_scrambling_control – This 2-bit field indicates the scrambling mode of the transport stream packet payload. The transport stream packet header, and the adaptation field when present, shall not be scrambled. In the case of a null packet the value of the transport_scrambling_control field shall be set to '00' (see Table 2-4).

Table 2-4 – Scrambling control values

Value	Description
'00'	Not scrambled
'01'	User-defined
'10'	User-defined
'11'	User-defined

adaptation_field_control – This 2-bit field indicates whether this transport stream packet header is followed by an adaptation field and/or payload (see Table 2-5).

Table 2-5 – Adaptation field control values

Value	Description
'00'	Reserved for future use by ISO/IEC
'01'	No adaptation_field, payload only
'10'	Adaptation_field only, no payload
'11'	Adaptation_field followed by payload

Rec. ITU-T H.222.0 | ISO/IEC 13818-1 decoders shall discard transport stream packets with the adaptation_field_control field set to a value of '00'. In the case of a null packet the value of the adaptation_field_control shall be set to '01'.

continuity_counter – The continuity_counter is a 4-bit field incrementing with each transport stream packet with the same PID. The continuity_counter wraps around to 0 after its maximum value. The continuity_counter shall not be incremented when the adaptation_field_control of the packet equals '00' or '10'.

In transport streams, duplicate packets may be sent as two, and only two, consecutive transport stream packets of the same PID. The duplicate packets shall have the same continuity_counter value as the original packet and the adaptation_field_control field shall be equal to '01' or '11'. In duplicate packets each byte of the original packet shall be duplicated, with the exception that in the program clock reference fields, if present, a valid value shall be encoded.

The continuity_counter in a particular transport stream packet is continuous when it differs by a positive value of one from the continuity_counter value in the previous transport stream packet of the same PID, or when either of the non-incrementing conditions (adaptation_field_control set to '00' or '10', or duplicate packets as described above) are met. The continuity counter may be discontinuous when the discontinuity_indicator is set to '1' (refer to 2.4.3.4). In the case of a null packet the value of the continuity_counter is undefined.

data_byte – Data bytes shall be contiguous bytes of data from the PES packets (refer to 2.4.3.6), transport stream sections (refer to 2.4.4), packet stuffing bytes after transport stream sections, or private data not in these structures as indicated by the PID. In the case of null packets with PID value 0x1FFF, data_bytes may be assigned any value. The number of data_bytes, N, is specified by 184 minus the number of bytes in the adaptation_field(), as described in 2.4.3.4.

2.4.3.4 Adaptation field

See Table 2-6.

Table 2-6 – Transport stream adaptation field

Syntax	No. of bits	Mnemonic
adaptation_field() {		
adaptation_field_length	**8**	**uimsbf**
if (adaptation_field_length > 0) {		
discontinuity_indicator	**1**	**bslbf**
random_access_indicator	**1**	**bslbf**
elementary_stream_priority_indicator	**1**	**bslbf**
PCR_flag	**1**	**bslbf**
OPCR_flag	**1**	**bslbf**
splicing_point_flag	**1**	**bslbf**
transport_private_data_flag	**1**	**bslbf**
adaptation_field_extension_flag	**1**	**bslbf**
if (PCR_flag = = '1') {		
program_clock_reference_base	**33**	**uimsbf**
reserved	**6**	**bslbf**
program_clock_reference_extension	**9**	**uimsbf**
}		
if (OPCR_flag = = '1') {		
original_program_clock_reference_base	**33**	**uimsbf**
reserved	**6**	**bslbf**
original_program_clock_reference_extension	**9**	**uimsbf**
}		
if (splicing_point_flag = = '1') {		
splice_countdown	**8**	**tcimsbf**
}		
if (transport_private_data_flag = = '1') {		
transport_private_data_length	**8**	**uimsbf**
for (i = 0; i < transport_private_data_length; i++) {		
private_data_byte	**8**	**bslbf**
}		
}		
if (adaptation_field_extension_flag = = '1') {		
adaptation_field_extension_length	**8**	**uimsbf**
ltw_flag	**1**	**bslbf**
piecewise_rate_flag	**1**	**bslbf**
seamless_splice_flag	**1**	**bslbf**
af_descriptor_not_present_flag	**1**	**bslbf**
reserved	**4**	**bslbf**
if (ltw_flag = = '1') {		
ltw_valid_flag	**1**	**bslbf**
ltw_offset	**15**	**uimsbf**
}		
if (piecewise_rate_flag = = '1') {		
reserved	**2**	**bslbf**
piecewise_rate	**22**	**uimsbf**
}		
if (seamless_splice_flag = = '1') {		
Splice_type	**4**	**bslbf**
DTS_next_AU[32..30]	**3**	**bslbf**
marker_bit	**1**	**bslbf**
DTS_next_AU[29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
DTS_next_AU[14..0]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
}		
if (af_descriptor_not_present_flag = = '0') {		
for (i = 0; i < N1; i++) {		
af_descriptor()		
}		
}		
else {		
for (i = 0; i < N2; i++) {		
reserved	**8**	**bslbf**
}		
}		
}		
for (i = 0; i < N3; i++) {		
stuffing_byte	**8**	**bslbf**
}		
}		
}		

2.4.3.5 Semantic definition of fields in adaptation field

adaptation_field_length – The adaptation_field_length is an 8-bit field specifying the number of bytes in the adaptation_field immediately following the adaptation_field_length. The value '0' is for inserting a single stuffing byte in the adaptation field of a transport stream packet. When the adaptation_field_control value is '11', the value of the adaptation_field_length shall be in the range 0 to 182. When the adaptation_field_control value is '10', the value of the adaptation_field_length shall be 183. For transport stream packets carrying PES packets, stuffing is needed when there is insufficient PES packet data to completely fill the transport stream packet payload bytes. Stuffing is accomplished by defining an adaptation field longer than the sum of the lengths of the data elements in it, so that the payload bytes remaining after the adaptation field exactly accommodates the available PES packet data. The extra space in the adaptation field is filled with stuffing bytes.

This is the only method of stuffing allowed for transport stream packets carrying PES packets. For transport stream packets carrying sections, an alternative stuffing method is described in 2.4.4.

discontinuity_indicator – This is a 1-bit field which when set to '1' indicates that the discontinuity state is true for the current transport stream packet. When the discontinuity_indicator is set to '0' or is not present, the discontinuity state is false. The discontinuity indicator is used to indicate two types of discontinuities, system time-base discontinuities and continuity_counter discontinuities.

A system time-base discontinuity is indicated by the use of the discontinuity_indicator in transport stream packets of a PID designated as a PCR_PID (refer to 2.4.4.9). When the discontinuity state is true for a transport stream packet of a PID designated as a PCR_PID, the next PCR in a transport stream packet with that same PID represents a sample of a new system time clock for the associated program. The system time-base discontinuity point is defined to be the instant in time when the first byte of a packet containing a PCR of a new system time-base arrives at the input of the T-STD. The discontinuity_indicator shall be set to '1' in the packet in which the system time-base discontinuity occurs. The discontinuity_indicator bit may also be set to '1' in transport stream packets of the same PCR_PID prior to the packet which contains the new system time-base PCR. In this case, once the discontinuity_indicator has been set to '1', it shall continue to be set to '1' in all transport stream packets of the same PCR_PID up to and including the transport stream packet which contains the first PCR of the new system time-base. After the occurrence of a system time-base discontinuity, no fewer than two PCRs for the new system time-base shall be received before another system time-base discontinuity can occur. Further, except when trick mode status is true, data from no more than two system time-bases shall be present in the set of T-STD buffers for one program at any time.

Prior to the occurrence of a system time-base discontinuity, the first byte of a transport stream packet which contains a PTS or DTS which refers to the new system time-base shall not arrive at the input of the T-STD. After the occurrence of a system time-base discontinuity, the first byte of a transport stream packet which contains a PTS or DTS which refers to the previous system time-base shall not arrive at the input of the T-STD.

A continuity_counter discontinuity is indicated by the use of the discontinuity_indicator in any transport stream packet. When the discontinuity state is true in any transport stream packet of a PID not designated as a PCR_PID, the continuity_counter in that packet may be discontinuous with respect to the previous transport stream packet of the same PID. When the discontinuity state is true in a transport stream packet of a PID that is designated as a PCR_PID, the continuity_counter may only be discontinuous in the packet in which a system time-base discontinuity occurs. A continuity counter discontinuity point occurs when the discontinuity state is true in a transport stream packet and the continuity_counter in the same packet is discontinuous with respect to the previous transport stream packet of the same PID. A continuity counter discontinuity point shall occur at most one time from the initiation of the discontinuity state until the conclusion of the discontinuity state. Furthermore, for all PIDs that are not designated as PCR_PIDs, when the discontinuity_indicator is set to '1' in a packet of a specific PID, the discontinuity_indicator may be set to '1' in the next transport stream packet of that same PID, but shall not be set to '1' in three consecutive transport stream packet of that same PID.

For the purpose of this clause, an elementary stream access point is defined as follows:

- ISO/IEC 11172-2 video and Rec. ITU-T H.262 | ISO/IEC 13818-2 video – The first byte of a video sequence header.
- ISO/IEC 14496-2 visual – The first byte of the visual object sequence header.
- AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 – The first byte of an AVC access unit. The SPS and PPS parameter sets referenced in this and all subsequent AVC access units in the coded video stream shall be provided after this access point in the byte stream and prior to their activation.
- Video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10 – The first byte of an SVC dependency representation is an elementary stream access point if the following conditions are met:

  - The subset sequence parameter sets and picture parameter sets referenced in this and all subsequent SVC dependency representation in the video sub-bitstream shall be provided after this access point in the byte stream and prior to their activation.
  - If this SVC video sub-bitstream access point requires the elementary stream access point of the same AVC access unit, if any, contained in the corresponding elementary stream that needs to be present in decoding order before decoding the elementary stream associated with this elementary stream access point, then the corresponding elementary stream shall also include an elementary stream access point.

    NOTE 1 – If the hierarchy descriptor is present for this SVC video sub-bitstream then the video sub-bitstream of which the hierarchy_layer_index equals the hierarchy_embedded_layer_index of this SVC sub-bitstream should have an elementary stream access point in the same access unit.

- MVC video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10 – The first byte of an MVC view-component subset is an elementary stream access point if the following two conditions are met:
  – The subset sequence parameter sets and picture parameter sets referenced in this and all subsequent MVC view-component subsets in the MVC video sub-bitstream shall be provided after this access point in the byte stream and prior to their activation.
  – If this MVC video sub-bitstream access point requires the elementary stream access point of the same AVC access unit, if any, contained in the corresponding elementary stream that needs to be present in decoding order before decoding the elementary stream associated with this elementary stream access point, then the corresponding elementary stream shall also include an elementary stream access point.

    NOTE 2 – If the hierarchy descriptor is present for this MVC video sub-bitstream, then the MVC video sub-bitstream of which the hierarchy_layer_index equals the hierarchy_embedded_layer_index of this MVC sub-bitstream should have an elementary stream access point in this same access unit.

- Audio – The first byte of an audio frame.
- ISO/IEC 14496-17 text stream – The first byte of a text access unit. In case in-band sample descriptions are used, each in-band sample description shall be provided in the ISO/IEC 14496-17 stream after this access point and prior to its use by an access unit.
- HEVC video streams or HEVC temporal video sub-bitstreams – The first byte of an HEVC access unit. The VPS, SPS and PPS parameter sets, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2, referenced in this and all subsequent HEVC access units in the HEVC video sequence shall be provided after this access point in the byte stream and prior to their activation.
- MVCD video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10 – The first byte of an MVCD view-component subset is an elementary stream access point if the following two conditions are met:
  – The subset sequence parameter sets and picture parameter sets referenced in this and all subsequent MVCD view-component subsets in the MVCD video sub-bitstream shall be provided after this access point in the byte stream and prior to their activation.
  – If this MVCD video sub-bitstream access point requires the elementary stream access point of the same AVC access unit, if any, contained in the corresponding elementary stream that needs to be present in decoding order before decoding the elementary stream associated with this elementary stream access point, then the corresponding elementary stream shall also include an elementary stream access point.

    NOTE 3 – If the hierarchy descriptor is present for this MVCD video sub-bitstream, then the MVCD video sub-bitstream of which the hierarchy_layer_index equals the hierarchy_embedded_layer_index of this MVCD sub-bitstream should have an elementary stream access point in this same access unit.

After a continuity counter discontinuity in a transport packet which is designated as containing elementary stream data, the first byte of elementary stream data in a transport stream packet of the same PID shall be the first byte of an elementary stream access point. In the case of ISO/IEC 11172-2, or Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 14496-2 video, the first byte of an elementary stream access point may also be the first byte of a sequence_end_code followed by an elementary stream access point.

Each transport stream packet which contains elementary stream data with a PID not designated as a PCR_PID, and in which a continuity counter discontinuity point occurs, and in which a PTS or DTS occurs, shall arrive at the input of the T-STD after the system time-base discontinuity for the associated program occurs. In the case where the discontinuity state is true, if two consecutive transport stream packets of the same PID occur which have the same continuity_counter value and have adaptation_field_control values set to '01' or '11', the second packet may be discarded. A transport stream shall not be constructed in such a way that discarding such a packet will cause the loss of PES packet payload data or transport stream section data.

After the occurrence of a discontinuity_indicator set to '1' in a transport stream packet which contains PSI information, a single discontinuity in the version_number of PSI sections may occur. At the occurrence of such a discontinuity, a version of the TS_program_map_sections of the appropriate program shall be sent with section_length = = 13 and the current_next_indicator = = 1, such that there are no program_descriptors and no elementary streams described. This shall then be followed by a version of the TS_program_map_section for each affected program with the version_number incremented by one and the current_next_indicator = = 1, containing a complete program definition. This indicates a version change in PSI data.

random_access_indicator – The random_access_indicator is a 1-bit field that indicates that the current transport stream packet, and possibly subsequent transport stream packets with the same PID, contain some information to aid random access at this point.

Specifically, when the bit is set to '1', the next PES packet to start in the payload of transport stream packets with the current PID shall contain an elementary stream access point as defined in the semantics for the discontinuity_indicator field. In addition, in the case of video, a presentation timestamp shall be present for the first picture following the elementary stream access point.

In the case of audio, the presentation timestamp shall be present in the PES packet containing the first byte of the audio frame. In the PCR_PID the random_access_indicator may only be set to '1' in a transport stream packet containing the PCR fields.

elementary_stream_priority_indicator – The elementary_stream_priority_indicator is a 1-bit field. It indicates, among packets with the same PID, the priority of the elementary stream data carried within the payload of this transport stream packet. A '1' indicates that the payload has a higher priority than the payloads of other transport stream packets.

In the case of ISO/IEC 11172-2 or Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 14496-2 video, this field may be set to '1' only if the payload contains one or more bytes from an intra-coded slice.

In the case of Rec. ITU-T H.264 | ISO/IEC 14496-10 video, this field may be set to '1' only if the payload contains one or more bytes from a slice with slice_type set to 2, 4, 7, or 9.

A value of '0' indicates that the payload has the same priority as all other packets which do not have this bit set to '1'.

For MVC video sub-bitstreams or MVC base view sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, this field may be set to '1' only if the payload contains one or more bytes from an anchor picture, indicated by the slice_type equal to 2, 4, 7, or 9 and the anchor_pic_flag syntax element equal to 1 for all prefix NAL units and slice extension NAL units.

For MVCD video sub-bitstreams or MVCD base view sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, this field may be set to '1' only if the payload contains one or more bytes from an anchor picture, indicated by the slice_type equal to 2, 4, 7, or 9 and the anchor_pic_flag syntax element equal to 1 for all prefix NAL units and slice extension NAL units. In the case of HEVC video streams or HEVC temporal video sub-bitstreams or HEVC temporal video subsets, this field may be set to '1' only if the payload contains one or more bytes from a slice with slice_type set to 2. A value of '0' indicates that the payload has the same priority as all other packets which do not have this bit set to '1'.

PCR_flag – The PCR_flag is a 1-bit flag. A value of '1' indicates that the adaptation_field contains a PCR field coded in two parts. A value of '0' indicates that the adaptation field does not contain any PCR field.

OPCR_flag – The OPCR_flag is a 1-bit flag. A value of '1' indicates that the adaptation_field contains an OPCR field coded in two parts. A value of '0' indicates that the adaptation field does not contain any OPCR field.

splicing_point_flag – The splicing_point_flag is a 1-bit flag. When set to '1', a splice_countdown field shall be present in this adaptation field, specifying the occurrence of a splicing point. A value of '0' indicates that a splice_countdown field is not present in the adaptation field.

transport_private_data_flag – The transport_private_data_flag is a 1-bit flag. A value of '1' indicates that the adaptation field contains one or more private_data bytes. A value of '0' indicates the adaptation field does not contain any private_data bytes.

adaptation_field_extension_flag – The adaptation_field_extension_flag is a 1-bit field which when set to '1' indicates the presence of an adaptation field extension. A value of '0' indicates that an adaptation field extension is not present in the adaptation field.

program_clock_reference_base; program_clock_reference_extension – The program_clock_reference (PCR) is a 42-bit field coded in two parts. The first part, program_clock_reference_base, is a 33-bit field whose value is given by

PCR_base(i), as given in equation 2-2. The second part, program_clock_reference_extension, is a 9-bit field whose value is given by PCR_ext(i), as given in equation 2-3. The PCR indicates the intended time of arrival of the byte containing the last bit of the program_clock_reference_base at the input of the system target decoder.

original_program_clock_reference_base; original_program_clock_reference_extension – The optional original program reference (OPCR) is a 42-bit field coded in two parts. These two parts, the base and the extension, are coded identically to the two corresponding parts of the PCR field. The presence of the OPCR is indicated by the OPCR_flag. The OPCR field shall be coded only in transport stream packets in which the PCR field is present. OPCRs are permitted in both single program and multiple program transport streams.

OPCR assists in the reconstruction of a single program transport stream from another transport stream. When reconstructing the original single program transport stream, the OPCR may be copied to the PCR field. The resulting PCR value is valid only if the original single program transport stream is reconstructed exactly in its entirety. This would include at least any PSI and private data packets which were present in the original transport stream and would possibly require other private arrangements. It also means that the OPCR must be an identical copy of its associated PCR in the original single program transport stream.

The OPCR is expressed as follows:

$$OPCR(i) = OPCR\_base(i) \times 300 + OPCR\_ext(i) \quad (2\text{-}8)$$

where:

$$OPCR\_base(i) = ((system\_clock\_frequency \times t(i)) DIV 300) \% 2^{33} \quad (2\text{-}9)$$

$$OPCR\_ext(i) = ((system\_clock\_frequency \times t(i)) DIV 1) \% 300 \quad (2\text{-}10)$$

The OPCR field is ignored by the decoder. The OPCR field shall not be modified by any multiplexor or decoder.

splice_countdown – The splice_countdown is an 8-bit field, representing a value which may be positive or negative. A positive value specifies the remaining number of transport stream packets, of the same PID, following the associated transport stream packet until a splicing point is reached. Duplicate transport stream packets and transport stream packets which only contain adaptation fields are excluded. The splicing point is located immediately after the last byte of the transport stream packet in which the associated splice_countdown field reaches zero. In the transport stream packet where the splice_countdown reaches zero, the last data byte of the transport stream packet payload shall be the last byte of a coded audio frame or a coded picture. In the case of video, the corresponding access unit may or may not be terminated by a sequence_end_code. Transport stream packets with the same PID, which follow, may contain data from a different elementary stream of the same type.

The payload of the next transport stream packet of the same PID (duplicate packets and packets without payload being excluded) shall commence with the first byte of a PES packet. In the case of audio, the PES packet payload shall commence with an access point. In the case of video, the PES packet payload shall commence with an access point, or with a sequence_end_code, followed by an access point. Thus, the previous coded audio frame or coded picture aligns with the packet boundary, or is padded to make this so. Subsequent to the splicing point, the countdown field may also be present. When the splice_countdown is a negative number whose value is minus n (–n), it indicates that the associated transport stream packet is the n-th packet following the splicing point (duplicate packets and packets without payload being excluded).

For the definition of an elementary stream access point, see the semantics of discontinuity_indicator.

transport_private_data_length – The transport_private_data_length is an 8-bit field specifying the number of private_data bytes immediately following the transport private_data_length field. The number of private_data bytes shall be such that private data does not extend beyond the adaptation field.

private_data_byte – The private_data_byte is an 8-bit field that shall not be specified by ITU-T | ISO/IEC.

adaptation_field_extension_length – The adaptation_field_extension_length is an 8-bit field. It indicates the number of bytes of the extended adaptation field data immediately following this field, including reserved bytes if present.

ltw_flag (legal time window_flag) – This is a 1-bit field which when set to '1' indicates the presence of the ltw_offset field.

piecewise_rate_flag – This is a 1-bit field which when set to '1' indicates the presence of the piecewise_rate field.

seamless_splice_flag – This is a 1-bit flag which when set to '1' indicates that the splice_type and DTS_next_AU fields are present. A value of '0' indicates that neither splice_type nor DTS_next_AU fields are present. This field shall be set to '0' in transport stream packets in which the splicing_point_flag is set to '0'. Once it is set to '1' in a transport stream packet in which the splice_countdown is positive, it shall be set to '1' in all the subsequent transport stream packets of the same PID that have the splicing_point_flag set to '1', until the packet in which the splice_countdown reaches zero (including this packet).

When this flag is set, and if the elementary stream carried in this PID is not a Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream, then the splice_type field shall be set to '0000'. If the elementary stream carried in this PID is an Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream, it shall fulfil the constraints indicated by the splice_type value.

af_descriptor_not_present_flag – This 1-bit field when set to '0' signals the presence of one or several af_descriptor() constructs in the adaptation header. When this flag is set to '1' it indicates that the af_descriptor() is not present in the adaptation header.

ltw_valid_flag (legal time window_valid_flag) – This is a 1-bit field which when set to '1' indicates that the value of the ltw_offset shall be valid. A value of '0' indicates that the value in the ltw_offset field is undefined.

ltw_offset (legal time window offset) – This is a 15-bit field, the value of which is defined only if the ltw_valid flag has a value of '1'. When defined, the legal time window offset is in units of ($300/f_s$) seconds, where f_s is the system clock frequency of the program that this PID belongs to, and fulfils:

$$offset = t_1(i) - t(i)$$

$$ltw\_offset = offset \,//\, 1$$

where i is the index of the first byte of this transport stream packet, offset is the value encoded in this field, t(i) is the arrival time of byte i in the T-STD, and $t_1(i)$ is the upper bound in time of a time interval called the legal time window which is associated with this transport stream packet.

The legal time window has the property that if this transport stream is delivered to a T-STD starting at time $t_1(i)$, i.e., at the end of its legal time window, and all other transport stream packets of the same program are delivered at the end of their legal time windows, then:

- For video – The MB_n buffer for this PID in the T-STD shall contain less than 184 bytes of elementary stream data at the time the first byte of the payload of this transport stream packet enters it, and no buffer violations in the T-STD shall occur.
- For audio – The B_n buffer for this PID in the T-STD shall contain less than BS_{dec} + 1 bytes of elementary stream data at the time the first byte of this transport stream packet enters it, and no buffer violations in the T-STD shall occur.

Depending on factors including the size of the buffer MB_n and the rate of data transfer between MB_n and EB_n, it is possible to determine another time $t_0(i)$, such that if this packet is delivered anywhere in the interval [$t_0(i)$, $t_1(i)$], no T-STD buffer violations will occur. This time interval is called the legal time window. The value of t_0 is not defined in this Recommendation | International Standard.

The information in this field is intended for devices such as remultiplexers which may need this information in order to reconstruct the state of the buffers MB_n.

piecewise_rate – The meaning of this 22-bit field is only defined when both the ltw_flag and the ltw_valid_flag are set to '1'. When defined, it is a positive integer specifying a hypothetical bitrate R which is used to define the end times of the Legal Time Windows of transport stream packets of the same PID that follow this packet but do not include the legal_time_window_offset field.

Assume that the first byte of this transport stream packet and the N following transport stream packets of the same PID have indices A_i, A_{i+1}, ..., A_{i+N}, respectively, and that the N latter packets do not have a value encoded in the field legal_time_window_offset. Then the values $t_1(A_{i+j})$ shall be determined by:

$$t_1(A_{1+j}) = t_1(A_i) + j \times 188 \times 8\ bits/byte/R$$

where j goes from 1 to N.

All packets between this packet and the next packet of the same PID to include a legal_time_window_offset field shall be treated as if they had the value:

$$offset = t_1(A_i) - t(A_i)$$

corresponding to the value $t_1(.)$ as computed by the formula above encoded in the legal_time_window_offset field. t(j) is the arrival time of byte j in the T-STD.

The meaning of this field is not defined when it is present in a transport stream packet with no legal_time_window_offset field.

splice_type – This is a 4-bit field. From the first occurrence of this field onwards, it shall have the same value in all the subsequent transport stream packets of the same PID in which it is present, until the packet in which the splice_countdown reaches zero (including this packet). If the elementary stream carried in that PID is not a Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream, then this field shall have the value '1111' (unspecified). If the elementary stream carried in that PID is a Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream, then this field indicates the conditions that shall be respected by this elementary stream for splicing purposes. These conditions are defined as a function of profile, level and splice_type in Table 2-7 through Table 2-20.

In these tables, a value for 'splice_decoding_delay' and 'max_splice_rate' means that the following conditions shall be satisfied by the video elementary stream:

1) The last byte of the coded picture ending in the transport stream packet in which the splice_countdown reaches zero shall remain in the VBV buffer of the VBV model for an amount of time equal to (splice_decoding_delay $t_{n+1} - t_n$), where for the purpose of this subclause:
   - n is the index of the coded picture ending in the transport stream packet in which the splice_countdown reaches zero, i.e., the coded picture referred to above.
   - t_n is defined in C.3.1 of Rec. ITU-T H.262 | ISO/IEC 13818-2.
   - ($t_{n+1} - t_n$) is defined in C.9 through C.12 of Rec. ITU-T H.262 | ISO/IEC 13818-2.

   NOTE – t_n is the time when coded picture n is removed from the VBV buffer, and ($t_{n+1} - t_n$) is the duration for which picture n is presented.

2) The VBV buffer of the VBV model shall not overflow if its input is switched at the splicing point to a stream of a constant rate equal to 'max_splice_rate' for an amount of time equal to 'splice_decoding_delay'.

Table 2-7 – Splice parameters Table 1
Simple Profile Main Level, Main Profile Main Level,
SNR Profile Main Level (both layers),
Spatial Profile High-1440 Level (base layer),
High Profile Main Level (middle + base layers),
Multi-view Profile Main Level (base layer) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 15.0×10^6 bit/s
'0001'	splice_decoding_delay = 150 ms; max_splice_rate = 12.0×10^6 bit/s
'0010'	splice_decoding_delay = 225 ms; max_splice_rate = 8.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 7.2×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-8 – Splice parameters Table 2
Main Profile Low Level, SNR Profile Low Level (both layers),
High Profile Main Level (base layer),
Multi-view Profile Low Level (base layer) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 115 ms; max_splice_rate = 4.0×10^6 bit/s
'0001'	splice_decoding_delay = 155 ms; max_splice_rate = 3.0×10^6 bit/s
'0010'	splice_decoding_delay = 230 ms; max_splice_rate = 2.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 1.8×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-9 – Splice parameters Table 3
Main Profile High-1440 Level, Spatial Profile High-1440 Level (all layers),
High Profile High-1440 Level (middle + base layers),
Multi-view Profile High-1440 Level (base layer) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 60.0×10^6 bit/s
'0001'	splice_decoding_delay = 160 ms; max_splice_rate = 45.0×10^6bit/s
'0010'	splice_decoding_delay = 240 ms; max_splice_rate = 30.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 28.5×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-10 – Splice parameters Table 4
Main Profile High Level, High Profile High-1440 Level (all layers),
High Profile High Level (middle + base layers),
Multi-view Profile High Level (base layer) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 80.0×10^6 bit/s
'0001'	splice_decoding_delay = 160 ms; max_splice_rate = 60.0×10^6 bit/s
'0010'	splice_decoding_delay = 240 ms; max_splice_rate = 40.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 38.0×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-11 – Splice parameters Table 5
SNR Profile Low Level (base layer) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 115 ms; max_splice_rate = 3.0×10^6 bit/s
'0001'	splice_decoding_delay = 175 ms; max_splice_rate = 2.0×10^6 bit/s
'0010'	splice_decoding_delay = 250 ms; max_splice_rate = 1.4×10^6 bit/s
'0011' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-12 – Splice parameters Table 6
SNR Profile Main Level (base layer) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 115 ms; max_splice_rate = 10.0×10^6 bit/s
'0001'	splice_decoding_delay = 145 ms; max_splice_rate = 8.0×10^6 bit/s
'0010'	splice_decoding_delay = 235 ms; max_splice_rate = 5.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 4.7×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-13 – Splice parameters Table 7
Spatial Profile High-1440 Level (middle + base layers) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 40.0×10^6 bit/s
'0001'	splice_decoding_delay = 160 ms; max_splice_rate = 30.0×10^6 bit/s
'0010'	splice_decoding_delay = 240 ms; max_splice_rate = 20.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 19.0×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-14 – Splice parameters Table 8
High Profile Main Level (all layers), High Profile High-1440 Level (base layer) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 20.0×10^6 bit/s
'0001'	splice_decoding_delay = 160 ms; max_splice_rate = 15.0×10^6 bit/s
'0010'	splice_decoding_delay = 240 ms; max_splice_rate = 10.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 9.5×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-15 – Splice parameters Table 9
High Profile High Level (base layer),
Multi-view Profile Main Level (both layers) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 25.0×10^6 bit/s
'0001'	splice_decoding_delay = 165 ms; max_splice_rate = 18.0×10^6 bit/s
'0010'	splice_decoding_delay = 250 ms; max_splice_rate = 12.0×10^6 bit/s
'0011' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-16 – Splice parameters Table 10
High Profile High Level (all layers),
Multi-view Profile High-1440 Level (both layers) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 100.0×10^6 bit/s
'0001'	splice_decoding_delay = 160 ms; max_splice_rate = 75.0×10^6 bit/s
'0010'	splice_decoding_delay = 240 ms; max_splice_rate = 50.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 48.0×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-17 – Splice parameters Table 11
4:2:2 Profile Main Level Video

splice_type	Conditions
'0000'	splice_decoding_delay = 45 ms; max_splice_rate = 50.0×10^6 bit/s
'0001'	splice_decoding_delay = 90 ms; max_splice_rate = 50.0×10^6 bit/s
'0010'	splice_decoding_delay = 180 ms; max_splice_rate = 50.0×10^6 bit/s
'0011'	splice_decoding_delay = 225 ms; max_splice_rate = 40.0×10^6 bit/s
'0100'	splice_decoding_delay = 250 ms; max_splice_rate = 36.0×10^6 bit/s
'0101' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-18 – Splice parameters Table 12
Multi-view Profile Low Level (both layers) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 115 ms; max_splice_rate = 8.0×10^6 bit/s
'0001'	splice_decoding_delay = 155 ms; max_splice_rate = 6.0×10^6 bit/s
'0010'	splice_decoding_delay = 230 ms; max_splice_rate = 4.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 3.7×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-19 – Splice parameters Table 13
Multi-view Profile High Level (both layers) Video

splice_type	Conditions
'0000'	splice_decoding_delay = 120 ms; max_splice_rate = 130.0×10^6 bit/s
'0001'	splice_decoding_delay = 150 ms; max_splice_rate = 104.0×10^6 bit/s
'0010'	splice_decoding_delay = 240 ms; max_splice_rate = 65.0×10^6 bit/s
'0011'	splice_decoding_delay = 250 ms; max_splice_rate = 62.4×10^6 bit/s
'0100' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

Table 2-20 – Splice parameters Table 14
4:2:2 Profile High Level Video

splice_type	Conditions
'0000'	splice_decoding_delay = 45 ms; max_splice_rate = 300.0×10^6 bit/s
'0001'	splice_decoding_delay = 90 ms; max_splice_rate = 300.0×10^6 bit/s
'0010' .. '0011'	Reserved
'0100'	splice_decoding_delay = 250 ms; max_splice_rate = 180.0×10^6 bit/s
'0101' .. '1011'	Reserved
'1100' .. '1110'	User-defined
'1111'	unspecified

DTS_next_AU (decoding time stamp next access unit) – This is a 33-bit field, coded in three parts. In the case of continuous and periodic decoding through this splicing point it indicates the decoding time of the first access unit following the splicing point. This decoding time is expressed in the time base which is valid in the transport stream packet in which the splice_countdown reaches zero. From the first occurrence of this field onwards, it shall have the same value in all the subsequent transport stream packets of the same PID in which it is present, until the packet in which the splice_countdown reaches zero (including this packet).

The af_descriptor() field may carry one or more descriptors as defined in Annex U. For descriptors carrying information associated with specific access units of an elementary stream, the descriptor applies to the first access unit that starts in the PES packet immediately following this adaptation field. There may be several TS packets carrying no payload before the start of the PES, in which case these descriptors apply to the next TS packet with payload on the same PID.

The adaptation field shall contain only complete af_descriptor() descriptors, i.e., a single descriptor is always contained in a single transport stream packet.

NOTE 5 – The adaptation field should remain relatively small; it is therefore recommended for large descriptors to use PES carriage as defined in Annex U.

stuffing_byte – This is a fixed 8-bit value equal to '1111 1111' that can be inserted by the encoder. It is discarded by the decoder.

2.4.3.6 PES packet

See Table 2-21.

Table 2-21 – PES packet

Syntax	No. of bits	Mnemonic
PES_packet() {		
packet_start_code_prefix	**24**	**bslbf**
stream_id	**8**	**uimsbf**
PES_packet_length	**16**	**uimsbf**
if (stream_id != program_stream_map		
&& stream_id != padding_stream		
&& stream_id != private_stream_2		
&& stream_id != ECM		
&& stream_id != EMM		
&& stream_id != program_stream_directory		
&& stream_id != DSMCC_stream		
&& stream_id != ITU-T Rec. H.222.1 type E stream) {		
'10'	**2**	**bslbf**
PES_scrambling_control	**2**	**bslbf**
PES_priority	**1**	**bslbf**
data_alignment_indicator	**1**	**bslbf**
copyright	**1**	**bslbf**
original_or_copy	**1**	**bslbf**
PTS_DTS_flags	**2**	**bslbf**
ESCR_flag	**1**	**bslbf**
ES_rate_flag	**1**	**bslbf**
DSM_trick_mode_flag	**1**	**bslbf**
additional_copy_info_flag	**1**	**bslbf**
PES_CRC_flag	**1**	**bslbf**
PES_extension_flag	**1**	**bslbf**
PES_header_data_length	**8**	**uimsbf**
if (PTS_DTS_flags == '10') {		
'0010'	**4**	**bslbf**
PTS [32..30]	**3**	**bslbf**
marker_bit	**1**	**bslbf**
PTS [29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
PTS [14..0]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
}		
if (PTS_DTS_flags == '11') {		
'0011'	**4**	**bslbf**
PTS [32..30]	**3**	**bslbf**
marker_bit	**1**	**bslbf**
PTS [29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
PTS [14..0]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
'0001'	**4**	**bslbf**
DTS [32..30]	**3**	**bslbf**
marker_bit	**1**	**bslbf**
DTS [29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
DTS [14..0]	**15**	**bslbf**

Syntax	No. of bits	Mnemonic
marker_bit	1	bslbf
}		
if (ESCR_flag == '1') {		
reserved	2	bslbf
ESCR_base[32..30]	3	bslbf
marker_bit	1	bslbf
ESCR_base[29..15]	15	bslbf
marker_bit	1	bslbf
ESCR_base[14..0]	15	bslbf
marker_bit	1	bslbf
ESCR_extension	9	uimsbf
marker_bit	1	bslbf
}		
if (ES_rate_flag == '1') {		
marker_bit	1	bslbf
ES_rate	22	uimsbf
marker_bit	1	bslbf
}		
if (DSM_trick_mode_flag == '1') {		
trick_mode_control	3	uimsbf
if (trick_mode_control == fast_forward) {		
field_id	2	bslbf
intra_slice_refresh	1	bslbf
frequency_truncation	2	bslbf
}		
else if (trick_mode_control == slow_motion) {		
rep_cntrl	5	uimsbf
}		
else if (trick_mode_control == freeze_frame) {		
field_id	2	uimsbf
reserved	3	bslbf
}		
else if (trick_mode_control == fast_reverse) {		
field_id	2	bslbf
intra_slice_refresh	1	bslbf
frequency_truncation	2	bslbf
else if (trick_mode_control == slow_reverse) {		
rep_cntrl	5	uimsbf
}		
else		
reserved	5	bslbf
}		
if (additional_copy_info_flag == '1') {		
marker_bit	1	bslbf
additional_copy_info	7	bslbf
}		
if (PES_CRC_flag == '1') {		
previous_PES_packet_CRC	16	bslbf
}		
if (PES_extension_flag == '1') {		
PES_private_data_flag	1	bslbf
pack_header_field_flag	1	bslbf
program_packet_sequence_counter_flag	1	bslbf
P-STD_buffer_flag	1	bslbf
reserved	3	bslbf
PES_extension_flag_2	1	bslbf
if (PES_private_data_flag == '1') {		
PES_private_data	128	bslbf
}		
if (pack_header_field_flag == '1') {		
pack_field_length	8	uimsbf
pack_header()		
}		
if (program_packet_sequence_counter_flag == '1') {		
marker_bit	1	bslbf
program_packet_sequence_counter	7	uimsbf
marker_bit	1	bslbf
MPEG1_MPEG2_identifier	1	bslbf
original_stuff_length	6	uimsbf
}		
if (P-STD_buffer_flag == '1') {		
'01'	2	bslbf
P-STD_buffer_scale	1	bslbf
P-STD_buffer_size	13	uimsbf

Syntax	No. of bits	Mnemonic
}		
if (PES_extension_flag_2 == '1') {		
marker_bit	**1**	**bslbf**
PES_extension_field_length	**7**	**uimsbf**
stream_id_extension_flag	**1**	**bslbf**
if (stream_id_extension_flag == '0') {		
stream_id_extension	**7**	**uimsbf**
}		
else {		
reserved	**6**	**bslbf**
tref_extension_flag	**1**	**bslbf**
if (tref_extension_flag == '0') {		
reserved	**4**	**bslbf**
TREF[32..30]	**3**	**bslbf**
marker_bit	**1**	**bslbf**
TREF[29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
TREF[14..0]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
}		
}		
for (i = 0; i < N3; i++) {		
reserved	**8**	**bslbf**
}		
}		
}		
for (i < 0; i < N1; i++) {		
stuffing_byte	**8**	**bslbf**
}		
for (i < 0; i < N2; i++) {		
PES_packet_data_byte	**8**	**bslbf**
}		
}		
else if (stream_id == program_stream_map		
|| stream_id == private_stream_2		
|| stream_id == ECM		
|| stream_id == EMM		
|| stream_id == program_stream_directory		
|| stream_id == DSMCC_stream		
|| stream_id == ITU-T Rec. H.222.1 type E stream) {		
for (i = 0; i < PES_packet_length; i++) {		
PES_packet_data_byte	**8**	**bslbf**
}		
}		
else if (stream_id == padding_stream) {		
for (i < 0; i < PES_packet_length; i++) {		
padding_byte	**8**	**bslbf**
}		
}		
}		
NOTE – The value N3 equals the byte count given by PES_extension_field_length minus the bytes which contain the stream_id_extension_flag and the occurring data elements in the if-else-construct after the stream_id_extension_flag.		

2.4.3.7 Semantic definition of fields in PES packet

packet_start_code_prefix – The packet_start_code_prefix is a 24-bit code. Together with the stream_id that follows it constitutes a packet start code that identifies the beginning of a packet. The packet_start_code_prefix is the bit string '0000 0000 0000 0000 0000 0001' (0x000001).

stream_id – In program streams, the stream_id specifies the type and number of the elementary stream as defined by the stream_id Table 2-22. In transport streams, the stream_id may be set to any valid value which correctly describes the elementary stream type as defined in Table 2-22. In transport streams, the elementary stream type is specified in the program-specific information as specified in 2.4.4. For AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, all video sub-bitstreams of the same AVC video stream shall have the same stream_id value. For AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, all MVC video sub-bitstreams of the same AVC video stream shall have the same stream_id value. For AVC video streams conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, all MVCD video sub-bitstreams of the same AVC video stream shall have the same stream_id value.

PES_packet_length – A 16-bit field specifying the number of bytes in the PES packet following the last byte of the field. A value of 0 indicates that the PES packet length is neither specified nor bounded and is allowed only in PES packets whose payload consists of bytes from a video elementary stream contained in transport stream packets.

PES_scrambling_control – The 2-bit PES_scrambling_control field indicates the scrambling mode of the PES packet payload. When scrambling is performed at the PES level, the PES packet header, including the optional fields when present, shall not be scrambled (see Table 2-23).

Table 2-22 – Stream_id assignments

Stream_id	Note	stream coding
'1011 1100'	1	program_stream_map
'1011 1101'	2,9,10	private_stream_1
'1011 1110'		padding_stream
'1011 1111'	3	private_stream_2
'110x xxxx'		ISO/IEC 13818-3 or ISO/IEC 11172-3 or ISO/IEC 13818-7 or ISO/IEC 14496-3 or ISO/IEC 23008-3 audio stream number 'x xxxx'
'1110 xxxx'		Rec. ITU-T H.262 | ISO/IEC 13818-2, ISO/IEC 11172-2, ISO/IEC 14496-2, Rec. ITU-T H.264 | ISO/IEC 14496-10 or Rec. ITU-T H.265 | ISO/IEC 23008-2 video stream number 'xxxx'
'1111 0000'	3	ECM_stream
'1111 0001'	3	EMM_stream
'1111 0010'	5	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Annex A or ISO/IEC 13818-6_DSMCC_stream
'1111 0011'	2	ISO/IEC_13522_stream
'1111 0100'	6	Rec. ITU-T H.222.1 type A
'1111 0101'	6	Rec. ITU-T H.222.1 type B
'1111 0110'	6	Rec. ITU-T H.222.1 type C
'1111 0111'	6	Rec. ITU-T H.222.1 type D
'1111 1000'	6	Rec. ITU-T H.222.1 type E
'1111 1001'	7	ancillary_stream
'1111 1010'		ISO/IEC 14496-1_SL-packetized_stream
'1111 1011'		ISO/IEC 14496-1_FlexMux_stream
'1111 1100'		metadata stream
'1111 1101'	8	extended_stream_id
'1111 1110'		reserved data stream
'1111 1111'	4	program_stream_directory

The notation x means that the values '0' or '1' are both permitted and results in the same stream type. The stream number is given by the values taken by the x's.

NOTE 1 – PES packets of type program_stream_map have unique syntax specified in 2.5.4.1.

NOTE 2 – PES packets of type private_stream_1 and ISO/IEC_13552_stream follow the same PES packet syntax as those for Rec. ITU-T H.262 | ISO/IEC 13818-2 video and ISO/IEC 13818-3 audio streams.

NOTE 3 – PES packets of type private_stream_2, ECM_stream and EMM_stream are similar to private_stream_1 except that no syntax is specified after PES_packet_length field.

NOTE 4 – PES packets of type program_stream_directory have a unique syntax specified in 2.5.5.

NOTE 5 – PES packets of type DSM-CC_stream have a unique syntax specified in ISO/IEC 13818-6.

NOTE 6 – This stream_id is associated with stream_type 0x09 in Table 2-34.

NOTE 7 – This stream_id is only used in PES packets, which carry data from a program stream or an ISO/IEC 11172-1 System Stream, in a transport stream (refer to 2.4.3.8).

NOTE 8 – The use of stream_id 0xFD (extended_stream_id) identifies that this PES packet employs an extended syntax to permit additional stream types to be identified.

NOTE 9 – JPEG 2000 video streams (stream_type = 0x21) are carried using the same PES packet syntax as private_stream_1.

NOTE 10 – Timeline and External Media Information streams (stream_type = 0x27) are carried using the same PES packet syntax as private_stream_1.

Table 2-23 – PES scrambling control values

Value	Description
'00'	Not scrambled
'01'	User-defined
'10'	User-defined
'11'	User-defined

PES_priority – This is a 1-bit field indicating the priority of the payload in this PES packet. A '1' indicates a higher priority of the payload of the PES packet payload than a PES packet payload with this field set to '0'. A multiplexor can use the PES_priority bit to prioritize its data within an elementary stream. This field shall not be changed by the transport mechanism.

data_alignment_indicator – This is a 1-bit flag. When set to a value of '1', it indicates that the PES packet header is immediately followed by the video syntax element or audio sync word indicated in the data_stream_alignment_descriptor in 2.6.10 if this descriptor is present. If set to a value of '1' and the descriptor is not present, alignment as indicated in alignment_type '01' in Table 2-53, Table 2-54 or Table 2-55 is required. When set to a value of '0', it is not defined whether any such alignment occurs or not.

copyright – This is a 1-bit field. When set to '1' it indicates that the material of the associated PES packet payload is protected by copyright. When set to '0' it is not defined whether the material is protected by copyright. A copyright descriptor described in 2.6.24 is associated with the elementary stream which contains this PES packet and the copyright flag is set to '1' if the descriptor applies to the material contained in this PES packet.

original_or_copy – This is a 1-bit field. When set to '1' the contents of the associated PES packet payload is an original. When set to '0' it indicates that the contents of the associated PES packet payload are a copy.

PTS_DTS_flags – This is a 2-bit field. When the PTS_DTS_flags field is set to '10', the PTS fields shall be present in the PES packet header. When the PTS_DTS_flags field is set to '11', both the PTS fields and DTS fields shall be present in the PES packet header. When the PTS_DTS_flags field is set to '00' no PTS or DTS fields shall be present in the PES packet header. The value '01' is forbidden.

ESCR_flag – A 1-bit flag, which when set to '1' indicates that ESCR base and extension fields are present in the PES packet header. When set to '0' it indicates that no ESCR fields are present.

ES_rate_flag – A 1-bit flag, which when set to '1' indicates that the ES_rate field is present in the PES packet header. When set to '0' it indicates that no ES_rate field is present.

DSM_trick_mode_flag – A 1-bit flag, which when set to '1' it indicates the presence of an 8-bit trick mode field. When set to '0' it indicates that this field is not present.

additional_copy_info_flag – A 1-bit flag, which when set to '1' indicates the presence of the additional_copy_info field. When set to '0' it indicates that this field is not present.

PES_CRC_flag – A 1-bit flag, which when set to '1' indicates that a CRC field is present in the PES packet. When set to '0' it indicates that this field is not present.

PES_extension_flag – A 1-bit flag, which when set to '1' indicates that an extension field exists in this PES packet header. When set to '0' it indicates that this field is not present.

PES_header_data_length – An 8-bit field specifying the total number of bytes occupied by the optional fields and any stuffing bytes contained in this PES packet header. The presence of optional fields is indicated in the byte that precedes the PES_header_data_length field.

marker_bit – A marker_bit is a 1-bit field that has the value '1'.

PTS (presentation time stamp) – Presentation times shall be related to decoding times as follows: The PTS is a 33-bit number coded in three separate fields. It indicates the time of presentation, $tp_n(k)$, in the system target decoder of a presentation unit k of elementary stream n. The value of PTS is specified in units of the period of the system clock frequency divided by 300 (yielding 90 kHz). The presentation time is derived from the PTS according to equation 2-11 below. Refer to 2.7.4 for constraints on the frequency of coding presentation timestamps.

$$PTS(k) = ((system\_clock\_frequency \times tp_n(k)) DIV\ 300) \%\ 2^{33} \qquad (2\text{-}11)$$

where $tp_n(k)$ is the presentation time of presentation unit $P_n(k)$.

In the case of audio, if a PTS is present in PES packet header it shall refer to the first access unit commencing in the PES packet. An audio access unit commences in a PES packet if the first byte of the audio access unit is present in the PES packet.

In the case of ISO/IEC 11172-2 video or ISO/IEC 14496-2 video, if a PTS is present in a PES packet header, it shall refer to the access unit containing the first picture start code that commences in this PES packet. A picture start code commences in a PES packet if the first byte of the picture start code is present in the PES packet. For I- and P-pictures in non-low_delay sequences and in the case when there is no decoding discontinuity between access units (AUs) k and k', the presentation time $t_{pn}(k)$ shall be equal to the decoding time $t_{dn}(k')$ of the next transmitted I- or P-picture (refer to 2.7.5). If there is a decoding discontinuity, or the stream ends, the difference between $t_{pn}(k)$ and $t_{dn}(k)$ shall be the same as if the original stream had continued without a discontinuity and without ending.

NOTE 1 – A low_delay sequence is an ISO/IEC 14496-2 video sequence in which the low_delay flag is set to '1' (refer to 6.2.3 of ISO/IEC 14496-2).

For Rec. ITU-T H.262 | ISO/IEC 13818-2 video, if a PTS is present in a PES packet header, it shall refer to the access unit containing the first picture start code that commences in this PES packet. A picture start code commences in a PES packet if the first byte of the picture start code is present in the PES packet. For I- and P-coded frames in non-low_delay sequences and in the case when there is no decoding discontinuity between access units (AUs) k and k', the presentation time $t_{pn}(k)$ shall be equal to the decoding time $t_{dn}(k')$ of the next transmitted I- or P-coded frame (refer to 2.7.5). If there is a decoding discontinuity, or the stream ends, the difference between $t_{pn}(k)$ and $t_{dn}(k)$ shall be the same as if the original stream had continued without a discontinuity and without ending.

NOTE 2 – A low_delay sequence is an Rec. ITU-T H.262 | ISO/IEC 13818-2 video sequence in which the low_delay flag is set to '1' (refer to 6.2.2.3 of Rec. ITU-T H.262 | ISO/IEC 13818-2). Also note that for field pictures the presentation time refers to the first field picture of the coded frame.

For AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 video, if a PTS is present in the PES packet header, it shall refer to the first AVC access unit that commences in this PES packet. An AVC access unit commences in a PES packet if the first byte of the AVC access unit is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each decoded AVC access unit, the PTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal DPB output time in the HRD, defined herein as $t_{o,n,dpb}(n) = t_{r,n}(n) + t_c$ * dpb_output_delay(n), where $t_{r,n}(n)$, t_c, and dpb_output_delay(n) are defined as in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

For video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, if a PTS is present in the PES packet header, it shall refer to the first SVC dependency representation that commences in this PES packet. An SVC dependency representation commences in a PES packet if the first byte of the SVC dependency representation is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each re-assembled and decoded AVC access unit, the PTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal DPB output time in the HRD, defined herein as $t_{o,n,dpb}(n) = t_{r,n}(n) + t_c$ * dpb_output_delay(n), where $t_{r,n}(n)$, t_c, and dpb_output_delay(n) are defined as in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE 3 – Different clocks may be used for derivation of PTS and $t_{o,n,dpb}(n)$.

For MVC video sub-bitstreams, MVC base view sub-bitstream or AVC video sub-bitstream of MVC of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, if a PTS is present in the PES packet header, it shall refer to the first MVC view-component subset that commences in this PES packet. An MVC view-component subset commences in a PES packet if the first byte of the MVC view-component subset is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each re-assembled and decoded AVC access unit, the PTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal DPB output time in the HRD, defined herein as $t_{o,n,dpb}(n) = t_{r,n}(n) + t_c$ * dpb_output_delay(n), where $t_{r,n}(n)$, t_c, and dpb_output_delay(n) are defined as in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

In the case of an ISO/IEC 14496-17 text stream, if a PTS is present in PES packet header, it shall refer to the first text access unit commencing in the PES packet. A text access unit commences in a PES packet if the first byte of the text access unit is present in the PES packet.

For MVCD video sub-bitstreams, MVCD base view sub-bitstream or AVC video sub-bitstream of MVCD of AVC video streams conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, if a PTS is present in the PES packet header, it shall refer to the first MVCD view-component subset that commences in this PES packet. An MVCD view-component subset commences in a PES packet if the first byte of the MVCD view-component subset is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each re-assembled and decoded AVC access unit, the PTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal DPB output time in the HRD, defined herein as $t_{o,n,dpb}(n) = t_{r,n}(n) + t_c * dpb\_output\_delay(n)$, where $t_{r,n}(n)$, t_c, and dpb_output_delay(n) are defined as in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

The presentation time $t_{pn}(k)$ shall be equal to the decoding time $t_{dn}(k)$ for:

- audio access units;
- access units in Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 14496-2 low delay video sequences;
- B-pictures in ISO/IEC 11172-2, Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 14496-2 video streams.
- text access units in ISO/IEC 14496-17.

If there is filtering in audio, it is assumed by the system model that filtering introduces no delay, hence the sample referred to by PTS at encoding is the same sample referred to by PTS at decoding. In the case of scalable coding refer to 2.7.6.

For HEVC video streams, HEVC temporal video sub-bitstreams and HEVC temporal video subsets, if a PTS is present in the PES packet header, it shall refer to the first HEVC access unit that commences in this PES packet. For HEVC video sub-partitions, if a PTS is present in the PES packet header, it shall refer to the first HEVC layer component that commences in this PES packet. An HEVC layer component commences in a PES packet if the first byte of the HEVC layer component is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2, for each HEVC access unit the PTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal DPB output time in the HRD, as defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2.

DTS (decoding time stamp) – The DTS is a 33-bit number coded in three separate fields. It indicates the decoding time, $td_n(j)$, in the system target decoder of an access unit j of elementary stream n. The value of DTS is specified in units of the period of the system clock frequency divided by 300 (yielding 90 kHz). The decoding time derived from the DTS according to equation 2-12 below:

$$DTS(j) = ((system\_clock\_frequency \times td_n(j)) DIV\ 300) \%\ 2^{33} \quad (2\text{-}12)$$

where $td_n(j)$ is the decoding time of access unit $A_n(j)$.

In the case of ISO/IEC 11172-2 video, Rec. ITU-T H.262 | ISO/IEC 13818-2 video, or ISO/IEC 14496-2 video, if a DTS is present in a PES packet header, it shall refer to the access unit containing the first picture start code that commences in this PES packet. A picture start code commences in a PES packet if the first byte of the picture start code is present in the PES packet.

For AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 video, if a DTS is present in the PES packet header, it shall refer to the first AVC access unit that commences in this PES packet. An AVC access unit commences in a PES packet if the first byte of the AVC access unit is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each AVC access unit the DTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal CPB removal time $t_{r,n}(n)$ in the HRD, as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

For video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, if a DTS is present in the PES packet header, it shall refer to the first SVC dependency representation that commences in this PES packet. An SVC dependency representation commences in a PES packet if the first byte of the SVC dependency representation is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each re-assembled AVC access unit the DTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal CPB removal time $t_{r,n}(n)$ in the HRD, as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

For MVC video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, if a DTS is present in the PES packet header, it shall refer to the first MVC view-component subset that commences in this PES packet. An MVC view-component subset commences in a PES packet if the first byte of the MVC view-component subset is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each re-assembled AVC access unit the DTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal CPB removal time $t_{r,n}(n)$ in the HRD, as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE 4 – Different clocks may be used for derivation of DTS and $t_{r,n}(n)$.

In the case of scalable coding refer to 2.7.6.

For HEVC video streams, HEVC temporal video sub-bitstreams and HEVC temporal video subsets, if a DTS is present in the PES packet header, it shall refer to the first HEVC access unit that commences in this PES packet. For HEVC video sub-partitions, if a DTS is present in the PES packet header, it shall refer to the first HEVC layer component that commences in this PES packet. An HEVC layer component commences in a PES packet if the first byte of the HEVC layer component is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2, for each HEVC access unit the DTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal CPB removal time in the HRD, as defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2.

For MVCD video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, if a DTS is present in the PES packet header, it shall refer to the first MVCD view-component subset that commences in this PES packet. An MVCD view-component subset commences in a PES packet if the first byte of the MVCD view-component subset is present in the PES packet. To achieve consistency between the STD model and the HRD model defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, for each re-assembled AVC access unit the DTS value in the STD shall, within the accuracy of their respective clocks, indicate the same instant in time as the nominal CPB removal time $t_{r,n}(n)$ in the HRD, as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

ESCR_base; ESCR_extension – The elementary stream clock reference is a 42-bit field coded in two parts. The first part, ESCR_base, is a 33-bit field whose value is given by ESCR_base(i), as given in equation 2-14. The second part, ESCR_ext, is a 9-bit field whose value is given by ESCR_ext(i), as given in equation 2-15. The ESCR field indicates the intended time of arrival of the byte containing the last bit of the ESCR_base at the input of the PES-STD for PES streams (refer to 2.5.2.4).

Specifically:

$$ESCR(i) = ESCR\_base(i) \times 300 + ESCR\_ext(i) \quad (2\text{-}13)$$

where:

$$ESCR\_base(i) = ((system\_clock\_frequency \times t(i)) DIV\ 300) \%\ 2^{33} \quad (2\text{-}14)$$

$$ESCR\_ext(i) = ((system\_clock\_frequency \times t(i)) DIV\ 1) \%\ 300 \quad (2\text{-}15)$$

The ESCR and ES_rate field (refer to semantics immediately following) contain timing information relating to the sequence of PES streams. These fields shall satisfy the constraints defined in 2.7.3.

ES_rate (elementary stream rate) – The ES_rate field is a 22-bit unsigned integer specifying the rate at which the system target decoder receives bytes of the PES packet in the case of a PES stream. The ES_rate is valid in the PES packet in which it is included and in subsequent PES packets of the same PES stream until a new ES_rate field is encountered. The value of the ES_rate is measured in units of 50 bytes/second. The value '0' is forbidden. The value of the ES_rate is used to define the time of arrival of bytes at the input of a P-STD for PES streams defined in 2.5.2.4. The value encoded in the ES_rate field may vary from PES_packet to PES_packet.

trick_mode_control – A 3-bit field that indicates which trick mode is applied to the associated video stream (see Table 2-24). In cases of other types of elementary streams, the meanings of this field and those defined by the following five bits are undefined. For the definition of trick_mode status, refer to the trick mode section of 2.4.2.4.

When trick_mode status is false, the number of times N, a picture is output by the decoding process for progressive sequences, is specified for each picture by the repeat_first_field and top_field_first fields in the case of Rec. ITU-T H.262 | ISO/IEC 13818-2 video, and is specified through the sequence header in the case of ISO/IEC 11172-2 Video.

For interlaced sequences, when trick_mode status is false, the number of times N, a picture is output by the decoding process for progressive sequences, is specified for each picture by the repeat_first_field and progressive_frame fields in the case of Rec. ITU-T H.262 | ISO/IEC 13818-2 video.

When trick mode status is true, the number of times that a picture shall be displayed depends on the value of N.

When the value of this field changes or trick mode operations cease, any combination of the following may occur:

- discontinuity in the time base;
- decoding discontinuity;
- continuity counter discontinuity.

Table 2-24 – Trick mode control values

Value	Description
'000'	Fast forward
'001'	Slow motion
'010'	Freeze frame
'011'	Fast reverse
'100'	Slow reverse
'101' .. '111'	Reserved

In the context of trick mode, the non-normal speed of decoding and presentation may cause the values of certain fields defined in video elementary stream data to be incorrect. Likewise, the semantic constraint on the slice structure may be invalid. The video syntax elements to which this exception applies are:

- bit_rate;
- vbv_delay;
- repeat_first_field;
- v_axis_positive;
- field_sequence;
- subcarrier;
- burst_amplitude;
- subcarrier_phase.

A decoder cannot rely on the values encoded in these fields when in trick mode.

Decoders are not normatively required to decode the trick_mode_control field. However, the following normative requirements shall apply to decoders that do decode the trick_mode_control field.

fast forward – The value '000', in the trick_mode_control field. When this value is present it indicates a fast forward video stream and defines the meaning of the following five bits in the PES packet header. The intra_slice_refresh bit may be set to '1' indicating that there may be missing macroblocks which the decoder may replace with co-sited macroblocks of previously decoded pictures. The field_id field, defined in Table 2-25, indicates which field or fields should be displayed. The frequency_truncation field indicates that a restricted set of coefficients may be included. The meaning of the values of this field are shown in Table 2-26.

slow motion – The value '001', in the trick_mode_control field. When this value is present it indicates a slow motion video stream and defines the meaning of the following five bits in the PES packet header. In the case of progressive sequences, the picture should be displayed N × rep_cntrl times, where N is defined above.

In the case of ISO/IEC 11172-2 video and Rec. ITU-T H.262 | ISO/IEC 13818-2 video progressive sequences, the picture should be displayed for N × rep_cntrl picture duration.

In the case of Rec. ITU-T H.262 | ISO/IEC 13818-2 interlaced sequences, the picture should be displayed for N × rep_cntrl field duration. If the picture is a frame picture, the first field to be displayed is the top field if top_field_first is 1, and the bottom field if top_field_first is '0' (refer to Rec. ITU-T H.262 | ISO/IEC 13818-2). This field is displayed for N × rep_cntrl / 2 field duration. The other field of the picture is then displayed for N – N × rep_cntrl / 2 field duration.

freeze frame – The value '010', in the trick_mode_control field. When this value is present it indicates a freeze frame video stream and defines the meaning of the following five bits in the PES packet header. The field_id field, defined in Table 2-25, identifies which field(s) should be displayed. The field_id field refers to the first video access unit that commences in the PES packet which contains the field_id field, unless the PES packet contains zero payload bytes. In the latter case the field_id field refers to the most recent previous video access unit.

fast reverse – The value '011', in the trick_mode_control field. When this value is present it indicates a fast reverse video stream and defines the meaning of the following five bits in the PES packet header. The intra_slice_refresh bit may be set to '1' indicating that there may be missing macroblocks which the decoder may replace with co-sited macroblocks of previously decoded pictures. The field_id field, defined in Table 2-25, indicates which field or fields should be displayed. The frequency_truncation field indicates that a restricted set of coefficients may be included. The meaning of the values of this field are shown in Table 2-26.

slow reverse – The value '100', in the trick_mode_control field. When this value is present it indicates a slow reverse video stream and defines the meaning of the following five bits in the PES packet header. In the case of

ISO/IEC 11172-2 video and Rec. ITU-T H.262 | ISO/IEC 13818-2 video progressive sequences, the picture should be displayed for N × rep_cntrl picture duration, where N is defined above.

In the case of Rec. ITU-T H.262 | ISO/IEC 13818-2 interlaced sequences, the picture should be displayed for N × rep_cntrl field duration. If the picture is a frame picture, the first field to be displayed is the bottom field if top_field_first is 1, and the top field if top_field_first is '0' (refer to Rec. ITU-T H.262 | ISO/IEC 13818-2). This field is displayed for N × rep_cntrl / 2 field duration. The other field of the picture is then displayed for N – N × rep_cntrl / 2 field duration.

field_id – A 2-bit field that indicates which field(s) should be displayed. It is coded according to Table 2-25.

Table 2-25 – Field_id field control values

Value	Description
'00'	Display from top field only
'01'	Display from bottom field only
'10'	Display complete frame
'11'	Reserved

intra_slice_refresh – A 1-bit flag, which when set to '1', indicates that there may be missing macroblocks between coded slices of video data in this PES packet. When set to '0' this may not occur. For more information, see Rec. ITU-T H.262 | ISO/IEC 13818-2. The decoder may replace missing macroblocks with co-sited macroblocks of previously decoded pictures.

frequency_truncation – A 2-bit field which indicates that a restricted set of coefficients may have been used in coding the video data in this PES packet. The values are defined in Table 2-26.

Table 2-26 – Coefficient selection values

Value	Description
'00'	Only DC coefficients are non-zero
'01'	Only the first three coefficients are non-zero
'10'	Only the first six coefficients are non-zero
'11'	All coefficients may be non-zero

rep_cntrl – A 5-bit field that indicates the number of times each field in an interlaced picture should be displayed, or the number of times that a progressive picture should be displayed. It is a function of the trick_mode_control field and the top_field_first bit in the video sequence header whether the top field or the bottom field should be displayed first in the case of interlaced pictures. The value '0' is forbidden.

additional_copy_info – This 7-bit field contains private data relating to copyright information.

previous_PES_packet_CRC – The previous_PES_packet_CRC is a 16-bit field that contains the CRC value that yields a zero output of the 16 registers in the decoder similar to the one defined in Annex A, but with the polynomial:

$$x^{16} + x^{12} + x^5 + 1$$

after processing the data bytes of the previous PES packet, exclusive of the PES packet header.

NOTE 5 – This CRC is intended for use in network maintenance such as isolating the source of intermittent errors. It is not intended for use by elementary stream decoders. It is calculated only over the data bytes because PES packet header data can be modified during transport.

PES_private_data_flag – A 1-bit flag which when set to '1' indicates that the PES packet header contains private data. When set to a value of '0' it indicates that private data is not present in the PES header.

pack_header_field_flag – A 1-bit flag which when set to '1' indicates that an ISO/IEC 11172-1 pack header or a program stream pack header is stored in this PES packet header. If this field is in a PES packet that is contained in a program stream, then this field shall be set to '0'. In a transport stream, when set to the value '0' it indicates that no pack header is present in the PES header.

program_packet_sequence_counter_flag – A 1-bit flag which when set to '1' indicates that the program_packet_sequence_counter, MPEG1_MPEG2_identifier, and original_stuff_length fields are present in this PES packet. When set to a value of '0' it indicates that these fields are not present in the PES header.

P-STD_buffer_flag – A 1-bit flag which when set to '1' indicates that the P-STD_buffer_scale and P-STD_buffer_size are present in the PES packet header. When set to a value of '0' it indicates that these fields are not present in the PES header.

PES_extension_flag_2 – A 1-bit field which when set to '1' indicates the presence of the PES_extension_field_length field and associated fields. When set to a value of '0' this indicates that the PES_extension_field_length field and any associated fields are not present.

PES_private_data – This is a 16-byte field which contains private data. This data, combined with the fields before and after, shall not emulate the packet_start_code_prefix (0x000001).

pack_field_length – This is an 8-bit field which indicates the length, in bytes, of the pack_header_field().

program_packet_sequence_counter – The program_packet_sequence_counter field is a 7-bit field. It is an optional counter that increments with each successive PES packet from a program stream or from an ISO/IEC 11172-1 stream or the PES packets associated with a single program definition in a transport stream, providing functionality similar to a continuity counter (refer to 2.4.3.2). This allows an application to retrieve the original PES packet sequence of a program stream or the original packet sequence of the original ISO/IEC 11172-1 stream. The counter will wrap around to 0 after its maximum value. Repetition of PES packets shall not occur. Consequently, no two consecutive PES packets in the program multiplex shall have identical program_packet_sequence_counter values.

MPEG1_MPEG2_identifier – A 1-bit flag which when set to '1' indicates that this PES packet carries information from an ISO/IEC 11172-1 stream. When set to '0' it indicates that this PES packet carries information from a program stream.

original_stuff_length – This 6-bit field specifies the number of stuffing bytes used in the original Rec. ITU-T H.222.0 | ISO/IEC 13818-1 PES packet header or in the original ISO/IEC 11172-1 packet header.

P-STD_buffer_scale – The P-STD_buffer_scale is a 1-bit field, the meaning of which is only defined if this PES packet is contained in a program stream. It indicates the scaling factor used to interpret the subsequent P-STD_buffer_size field. If the preceding stream_id indicates an audio stream, P-STD_buffer_scale shall have the value '0'. If the preceding stream_id indicates a video stream, P-STD_buffer_scale shall have the value '1'. For all other stream types, the value may be either '1' or '0'.

P-STD_buffer_size – The P-STD_buffer_size is a 13-bit unsigned integer, the meaning of which is only defined if this PES packet is contained in a program stream. It defines the size of the input buffer, BS_n, in the P-STD. If P-STD_buffer_scale has the value '0', then the P-STD_buffer_size measures the buffer size in units of 128 bytes. If P-STD_buffer_scale has the value '1', then the P-STD_buffer_size measures the buffer size in units of 1024 bytes. Thus:

$$\begin{gathered} \text{if } (P - STD\_buffer\_scale == 0) \\ BS_n = \bar{P} - STD\_buffer\_size \times 128 \end{gathered} \tag{2-16}$$

else:

$$BS_n = P - STD\_buffer\_size \times 1024 \tag{2-17}$$

The encoded value of the P-STD buffer size takes effect immediately when the P-STD_buffer_size field is received by the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 System Target Decoder (refer to 2.7.7).

For AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10, the size BS_n shall be larger than or equal to the size of the CPB signalled by the CpbSize[cpb_cnt_minus1] specified by the NAL hrd_parameters() in the AVC video stream. If the NAL hrd_parameters() are not present in the AVC video stream, then BS_n shall be larger than or equal to the size of the NAL CPB for the byte stream format defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 as 1200 × MaxCPB for the applied level.

For video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, the size BS_n shall be larger than or equal to the size of the CPB signalled by the CpbSize[cpb_cnt_minus1] specified by the NAL hrd_parameters() for the video sub-bitstream carried in elementary stream ES_n as defined in 2.14.3.6. If the NAL hrd_parameters() are not present in the video sub-bitstream, the size BS_n shall be larger than or equal to the size of the NAL CPB for the byte stream format defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 as 1200 × MaxCPB for the applied level for the elementary stream ES_n.

For MVC video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, the size BS_n shall be larger than or equal to the size of the CPB signalled by the CpbSize[cpb_cnt_minus1] specified by the NAL hrd_parameters() for the MVC video sub-bitstreams carried in elementary stream ES_n, as defined in 2.14.3.6. If the NAL hrd_parameters() are not present in the MVC video sub-bitstreams, the size BS_n shall be larger than or equal to the size of the NAL CPB for the byte stream format defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 as 1200 × MaxCPB for the applied level for the elementary stream ES_n.

PES_extension_field_length – This is a 7-bit field which specifies the length, in bytes, of the data following this field in the PES extension field up to and including any reserved bytes.

stream_id_extension_flag – A 1-bit flag, which when set to '0' indicates that a stream_id_extension field is present in the PES packet header. The value of '1' for this flag is reserved.

stream_id_extension – In program streams, the stream_id_extension specifies the type and number of the elementary stream as defined by the stream_id_extension in Table 2-27. In transport streams, the stream_id_extension may be set to any valid value which correctly describes the elementary stream type as defined in Table 2-27. In transport streams, the elementary stream type is specified in the program-specific information as specified in 2.4.4. Note that this field is used as an extension of the stream_id defined above. This field shall not be used unless the value of stream_id is 1111 1101.

Table 2-27 – Stream_id_extension assignments

stream_id_extension	Note	stream coding
'000 0000'	1	IPMP control information stream
'000 0001'	2	IPMP stream
'000 0010' .. '000 1111'		ISO/IEC 14496-17 text stream
'001 0000' .. '001 1111'		ISO/IEC 23002-3 auxiliary video stream
'010 0000' .. '011 1111'		reserved_data_stream
'100 0000' .. '111 1111'		private_stream
NOTE 1 – PES packets of stream_id_extension 0b000 0000 (IPMP control information stream) have a unique syntax specified in ISO/IEC 13818-11 (MPEG-2 IPMP). NOTE 2 – PES packets of stream_id_extension 0b000 0001 (IPMP stream) have a unique syntax specified in ISO/IEC 13818-11 (MPEG-2 IPMP).		

stuffing_byte – This is a fixed 8-bit value equal to '1111 1111' that can be inserted by the encoder, for example to meet the requirements of the channel. It is discarded by the decoder. No more than 32 stuffing bytes shall be present in one PES packet header.

PES_packet_data_byte – PES_packet_data_bytes shall be contiguous bytes of data from the elementary stream indicated by the packet's stream_id or PID. When the elementary stream data conforms to Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 13818-3, the PES_packet_data_bytes shall be byte aligned to the bytes of this Recommendation | International Standard. The byte-order of the elementary stream shall be preserved. The number of PES_packet_data_bytes, N, is specified by the PES_packet_length field. N shall be equal to the value indicated in the PES_packet_length minus the number of bytes between the last byte of the PES_packet_length field and the first PES_packet_data_byte.

In the case of a private_stream_1, private_stream_2, ECM_stream, or EMM_stream, the contents of the PES_packet_data_byte field are user definable and will not be specified by ITU-T | ISO/IEC in the future.

padding_byte – This is a fixed 8-bit value equal to '1111 1111'. It is discarded by the decoder.

tref_extension_flag – A 1-bit flag, which when set to '0' indicates that a TREF field is present in the PES packet header. The value of '1' for this flag is reserved.

TREF (timestamp reference) – The TREF is a 33-bit number coded in three separate fields. It indicates the decoding time value, $td_n(j)$, in the system target decoder as indicated by the DTS, or in absence of the DTS, by the PTS of the PES header of the same j-th access unit in a corresponding elementary stream n.

2.4.3.8 Carriage of program streams and ISO/IEC 11172-1 Systems streams in the transport stream

The transport stream contains optional fields to support the carriage of program streams and ISO/IEC 11172-1 Systems streams, in a way that allows simple reconstruction of the respective stream at the decoder.

When placing a program stream into a transport stream, program stream PES packets with stream_id values of private_stream_1, Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 video, and ISO/IEC 13818-3 or ISO/IEC 11172-3 audio, are carried in transport stream packets.

For these PES packets, when reconstructing the program stream at the transport stream decoder, the PES packet data is copied to the program stream being reconstructed.

For program streams PES packets with stream_id values of program_stream_map, padding_stream, private_stream_2, ECM, EMM, DSM_CC_stream, or program_stream_directory, all the bytes of the program stream PES packet, except for the packet_start_code_prefix, are placed into the data_bytes fields of a new PES packet. The stream_id of this new PES packet has the value of ancillary_stream (refer to Table 2-22). This new PES packet is then carried in transport stream packets.

When reconstructing the program stream at the transport stream decoder, for PES packets with a stream_id value of ancillary_stream_id, packet_start_code_prefix is written to the program stream being reconstructed, followed by the data_byte fields from these transport stream PES packets.

ISO/IEC 11172-1 streams are carried within transport streams by first replacing ISO/IEC 11172-1 packet headers with Rec. ITU-T H.262 | ISO/IEC 13818-2 PES packet headers. ISO/IEC 11172-1 packet header field values are copied to the equivalent Rec. ITU-T H.262 | ISO/IEC 13818-2 PES packet header fields.

The program_packet_sequence_counter field is included within the header of each PES packet carrying data from a program stream, or an ISO/IEC 11172-1 System stream. This allows the order of PES packets in the original program stream, or packets in the original ISO/IEC 11172-1 system stream, to be reproduced at the decoder.

The pack_header() field of a program stream, or an ISO/IEC 11172-1 system stream, is carried in the transport stream in the header of the immediately following PES packet.

2.4.4 Program-specific information

Program-specific information (PSI) includes both Rec. ITU-T H.222.0 | ISO/IEC 13818-1 normative data and private data that enable demultiplexing of programs by decoders. Programs are composed of one or more elementary streams, each labelled with a PID. Programs, elementary streams or parts thereof may be scrambled for conditional access. However, program-specific information shall not be scrambled.

In transport streams, program-specific information is classified into six table structures as shown in Table 2-28. While these structures may be thought of as simple tables, they shall be segmented into sections and inserted in transport stream packets, some with predetermined PIDs and others with user selectable PIDs.

Table 2-28 – Program-specific information

Structure Name	Stream Type	Reserved PID #	Description
Program association table	Rec. ITU-T H.222.0 | ISO/IEC 13818-1	0x00	Associates program number and program map table PID
Program map table	Rec. ITU-T H.222.0 | ISO/IEC 13818-1	Assigned in the PAT	Specifies PID values for components of one or more programs
Network information table	Private	Assigned in the PAT	Physical network parameters such as FDM frequencies, transponder numbers, etc.
Conditional access table	Rec. ITU-T H.222.0 | ISO/IEC 13818-1	0x01	Associates one or more (private) EMM streams each with a unique PID value
Transport stream description table	Rec. ITU-T H.222.0 | ISO/IEC 13818-1	0x02	Associates one or more descriptors from Table 2-45 to an entire transport stream
IPMP control information table	Rec. ITU-T H.222.0 | ISO/IEC 13818-1	0x03	Contains IPMP tool list, rights container, tool container defined in ISO/IEC 13818-11

Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined PSI tables shall be segmented into one or more sections that are carried within transports packets. A section is a syntactic structure that shall be used for mapping each Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined PSI table into transport stream packets.

Along with Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined PSI tables, it is possible to carry private data tables. The means by which private information is carried within transport stream packets is not defined by this Specification. It may be structured in the same manner used for carrying of Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined PSI tables, such that the syntax for mapping this private data is identical to that used for the mapping of Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined PSI tables. For this purpose, a private section is defined. If the private data is carried in transport stream packets with the same PID value as transport stream packets carrying program map tables (as identified in the program association table), then the private_section syntax and semantics shall be used. The data carried in the private_data_bytes may be scrambled. However, no other fields of the private_section shall be scrambled. This private_section allows data to be transmitted with a minimum of structure. When this structure is not used, the mapping of private data within transport stream packets is not defined by this Recommendation | International Standard.

Sections may be variable in length. The beginning of a section is indicated by a pointer_field in the transport stream packet payload. The syntax of this field is specified in Table 2-29.

Adaptation fields may occur in transport stream packets carrying transport stream sections.

Within a transport stream, packet stuffing bytes of value 0xFF may be found in the payload of transport stream packets carrying sections only after the last byte of a section. In this case all bytes until the end of the transport stream packet shall also be stuffing bytes of value 0xFF. These bytes may be discarded by a decoder. In such a case, the payload of the next transport stream packet with the same PID value shall begin with a pointer_field of value 0x00 indicating that the next section starts immediately thereafter.

Each transport stream shall contain one or more transport stream packets with PID value 0x0000. These transport stream packets together shall contain a complete program association table, providing a complete list of all programs within the transport stream. The most recently transmitted version of the table with the current_next_indicator set to a value of '1' shall always apply to the current data in the transport stream. Any changes in the programs carried within the transport stream shall be described in an updated version of the program association table carried in transport stream packets with PID value 0x0000. These sections shall all use table_id value 0x00. Only sections with this value of table_id are permitted within transport stream packets with PID value of 0x0000. For a new version of the PAT to become valid, all sections (as indicated in the last_section_number) with a new version_number and with the current_next_indicator set to '1' must exit B_{sys} defined in the T-STD (refer to 2.4.2.1). The PAT becomes valid when the last byte of the section needed to complete the table exits B_{sys}.

Whenever one or more elementary streams within a transport stream are scrambled, transport stream packets with a PID value 0x0001 shall be transmitted containing a complete conditional access table including CA_descriptors associated with the scrambled streams. The transmitted transport stream packets will together form one complete version of the conditional access table. The most recently transmitted version of the table with the current_next_indicator set to a value of '1' shall always apply to the current data in the transport stream. Any changes in scrambling making the existing table invalid or incomplete shall be described in an updated version of the conditional access table. These sections will all use table_id value 0x01. Only sections with this table_id value are permitted within transport stream packets with a PID value of 0x0001. For a new version of the CAT to become valid, all sections (as indicated in the last_section_number) with a new version_number and with the current_next_indicator set to '1' must exit B_{sys}. The CAT becomes valid when the last byte of the section needed to complete the table exits B_{sys}.

Each transport stream shall contain one or more transport stream packets with PID values which are labelled under the program association table as transport stream packets containing TS program map sections. Each program listed in the program association table shall be described in a unique TS_program_map_section. Every program shall be fully defined within the transport stream itself. Private data which has an associated elementary_PID field in the appropriate program map table section is part of the program. Other private data may exist in the transport stream without being listed in the program map table section. The most recently transmitted version of the TS_program_map_section with the current_next_indicator set to a value of '1' shall always apply to the current data within the transport stream. Any changes in the definition of any of the programs carried within the transport stream shall be described in an updated version of the corresponding section of the program map table carried in transport stream packets with the PID value identified as the program_map_PID for that specific program. All transport stream packets which carry a given TS_program_map_section shall have the same PID value. During the continuous existence of a program, including all of its associated events, the program_map_PID shall not change. A program definition shall not span more than one TS_program_map_section. A new version of a TS_program_map_section becomes valid when the last byte of that section with a new version_number and with the current_next_indicator set to '1' exits $B_{sys.}$

Sections with a table_id value of 0x02 shall contain program map table information. Such sections may be carried in transport stream packets with different PID values.

The network information table is optional and its contents are private. If present it is carried within transport stream packets that will have the same PID value, called the network_PID. The network_PID value is defined by the user and, when present, shall be found in the program association table under the reserved program_number 0x0000. If the network information table exists, it shall take the form of one or more private_sections.

The maximum number of bytes in a section of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined PSI table is 1024 bytes. The maximum number of bytes in a private_section is 4096 bytes.

The transport stream description table is optional. When present, the transport stream description is carried within transport stream packets that have a PID value 0x0002 as specified in Table 2-28 and shall apply to the entire transport stream. Sections of the transport stream description shall use a table_id value of 0x03 as specified in Table 2-31 and its contents are restricted to descriptors specified in Table 2-45. The TS_description_section becomes valid when the last byte of the section required to complete the table exits B_{sys}.

There are no restrictions on the occurrence of start codes, sync bytes or other bit patterns in transport stream section data.

2.4.4.1 Pointer

The pointer_field syntax is defined in Table 2-29.

Table 2-29 – Program-specific information pointer

Syntax	No. of bits	Mnemonic
pointer_field	**8**	**uimsbf**

2.4.4.2 Semantics definition of fields in pointer syntax

pointer_field – This is an 8-bit field whose value shall be the number of bytes, immediately following the pointer_field until the first byte of the first section that is present in the payload of the transport stream packet (so a value of 0x00 in the pointer_field indicates that the section starts immediately after the pointer_field). When at least one section begins in a given transport stream packet, then the payload_unit_start_indicator (refer to 2.4.3.2) shall be set to '1' and the first byte of the payload of that transport stream packet shall contain the pointer. When no section begins in a given transport stream packet, then the payload_unit_start_indicator shall be set to '0' and no pointer shall be sent in the payload of that packet.

2.4.4.3 Program association table

The program association table (PAT) provides the correspondence between a program_number and the PID value of the transport stream packets which carry the program definition. The program_number is the numeric label associated with a program.

The overall table is contained in one or more sections with the following syntax. It may be segmented to occupy multiple sections (see Table 2-30).

Table 2-30 – Program association section

Syntax	No. of bits	Mnemonic
program_association_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
'0'	**1**	**bslbf**
reserved	**2**	**bslbf**
section_length	**12**	**uimsbf**
transport_stream_id	**16**	**uimsbf**
reserved	**2**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
for (i = 0; i < N; i++) {		
program_number	**16**	**uimsbf**
reserved	**3**	**bslbf**
if (program_number = = '0') {		
network_PID	**13**	**uimsbf**
}		
else {		
program_map_PID	**13**	**uimsbf**
}		
}		
CRC_32	**32**	**rpchof**
}		

2.4.4.4 Table_id assignments

The table_id field identifies the contents of a transport stream section as shown in Table 2-31.

Table 2-31 – table_id assignment values

Value	Description
0x00	program_association_section
0x01	conditional_access_section (CA_section)
0x02	TS_program_map_section
0x03	TS_description_section
0x04	ISO_IEC_14496_scene_description_section
0x05	ISO_IEC_14496_object_descriptor_section
0x06	Metadata_section
0x07	IPMP Control Information Section (defined in ISO/IEC 13818-11)
0x08	ISO_IEC_14496_section
0x09	ISO/IEC 23001-11 (Green access unit) section
0x0A	ISO/IEC 23001-10 (Quality access unit) section
0x0B .. 0x37	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 reserved
0x38 .. 0x3F	Defined in ISO/IEC 13818-6
0x40 .. 0xFE	User private
0xFF	Forbidden

2.4.4.5 Semantic definition of fields in program association section

table_id – This is an 8-bit field, which shall be set to 0x00 as shown in Table 2-31.

section_syntax_indicator – The section_syntax_indicator is a 1-bit field which shall be set to '1'.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the section, starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

transport_stream_id – This is a 16-bit field which serves as a label to identify this transport stream from any other multiplex within a network. Its value is defined by the user.

version_number – This 5-bit field is the version number of the whole program association table. The version number shall be incremented by 1 modulo 32 whenever the definition of the program association table changes. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable program association table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable program association table.

current_next_indicator – A 1-bit indicator, which when set to '1' indicates that the program association table sent is currently applicable. When the bit is set to '0', it indicates that the table sent is not yet applicable and shall be the next table to become valid.

section_number – This 8-bit field gives the number of this section. The section_number of the first section in the Program Association Table shall be 0x00. It shall be incremented by 1 with each additional section in the program association table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the complete program association table.

program_number – Program_number is a 16-bit field. It specifies the program to which the program_map_PID is applicable. When set to 0x0000, then the following PID reference shall be the network PID. For all other cases the value of this field is user defined. This field shall not take any single value more than once within one version of the program association table.

NOTE – The program_number may be used as a designation for a broadcast channel, for example.

network_PID – The network_PID is a 13-bit field, which is used only in conjunction with the value of the program_number set to 0x0000, specifies the PID of the transport stream packets which shall contain the network information table. The value of the network_PID field is defined by the user, but shall only take values as specified in Table 2-3. The presence of the network_PID is optional.

program_map_PID – The program_map_PID is a 13-bit field specifying the PID of the transport stream packets which shall contain the program_map_section applicable for the program as specified by the program_number. No program_number shall have more than one program_map_PID assignment. The value of the program_map_PID is defined by the user, but shall only take values as specified in Table 2-3.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire program association section.

2.4.4.6 Conditional access table

The conditional access table (CAT) provides the association between one or more conditional access (CA) systems, their EMM streams and any special parameters associated with them. Refer to 2.6.16 for a definition of the descriptor() field in Table 2-32.

The table is contained in one or more sections with the following syntax. It may be segmented to occupy multiple sections.

Table 2-32 – Conditional access section

Syntax	No. of bits	Mnemonic
CA_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
'0'	**1**	**bslbf**
reserved	**2**	**bslbf**
section_length	**12**	**uimsbf**
reserved	**18**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
for (i = 0; i < N; i++) {		
descriptor()		
}		
CRC_32	**32**	**rpchof**
}		

2.4.4.7 Semantic definition of fields in conditional access section

table_id – This is an 8-bit field, which shall be set to 0x01 as specified in Table 2-31.

section_syntax_indicator – The section_syntax_indicator is a 1-bit field which shall be set to '1'.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10-bits specify the number of bytes of the section starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

version_number – This 5-bit field is the version number of the entire conditional access table. The version number shall be incremented by 1 modulo 32 when a change in the information carried within the CA table occurs. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable conditional access table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable conditional access table.

current_next_indicator – A 1-bit indicator, which when set to '1' indicates that the conditional access table sent is currently applicable. When the bit is set to '0', it indicates that the conditional access table sent is not yet applicable and shall be the next conditional access table to become valid.

section_number – This 8-bit field gives the number of this section. The section_number of the first section in the conditional access table shall be 0x00. It shall be incremented by 1 with each additional section in the conditional access table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the conditional access table.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire conditional access section.

2.4.4.8 Program map table

The program map table provides the mappings between program numbers and the program elements that comprise them. A single instance of such a mapping is referred to as a "program definition". The program map table is the complete collection of all program definitions for a transport stream. This table shall be transmitted in packets, the PID values of which are selected by the encoder. More than one PID value may be used, if desired. The table is contained in one or more sections with the following syntax. It may be segmented to occupy multiple sections. In each section, the section number field shall be set to zero. Sections are identified by the program_number field.

Definition for the descriptor() fields may be found in 2.6 (see Table 2-33).

Table 2-33 – Transport stream program map section

Syntax	No. of bits	Mnemonic
TS_program_map_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
'0'	**1**	**bslbf**
reserved	**2**	**bslbf**
section_length	**12**	**uimsbf**
program_number	**16**	**uimsbf**
reserved	**2**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
reserved	**3**	**bslbf**
PCR_PID	**13**	**uimsbf**
reserved	**4**	**bslbf**
program_info_length	**12**	**uimsbf**
for (i = 0; i < N; i++) {		
descriptor()		
}		
for (i = 0; i < N1; i++) {		
stream_type	**8**	**uimsbf**
reserved	**3**	**bslbf**
elementary_PID	**13**	**uimsbf**
reserved	**4**	**bslbf**
ES_info_length	**12**	**uimsbf**
for (i = 0; i < N2; i++) {		
descriptor()		
}		
}		
CRC_32	**32**	**rpchof**
}		

2.4.4.9 Semantic definition of fields in transport stream program map section

table_id – This is an 8-bit field, which in the case of a TS_program_map_section shall be always set to 0x02 as shown in Table 2-31.

section_syntax_indicator – The section_syntax_indicator is a 1-bit field which shall be set to '1'.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the section starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

program_number – program_number is a 16-bit field. It specifies the program to which the program_map_PID is applicable. One program definition shall be carried within only one TS_program_map_section. This implies that a program definition is never longer than 1016 (0x3F8). See Informative Annex C for ways to deal with the cases when that length is not sufficient. The program_number may be used as a designation for a broadcast channel, for example. By describing the different program elements belonging to a program, data from different sources (e.g., sequential events) can be concatenated together to form a continuous set of streams using a program_number. For examples of applications refer to Annex C.

version_number – This 5-bit field is the version number of the TS_program_map_section. The version number shall be incremented by 1 modulo 32 when a change in the information carried within the section occurs. Version number refers to the definition of a single program, and therefore to a single section. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable TS_program_map_section. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable TS_program_map_section.

current_next_indicator – A 1-bit field, which when set to '1' indicates that the TS_program_map_section sent is currently applicable. When the bit is set to '0', it indicates that the TS_program_map_section sent is not yet applicable and shall be the next TS_program_map_section to become valid.

section_number – The value of this 8-bit field shall be 0x00.

last_section_number – The value of this 8-bit field shall be 0x00.

PCR_PID – This is a 13-bit field indicating the PID of the transport stream packets which shall contain the PCR fields valid for the program specified by program_number. If no PCR is associated with a program definition for private streams, then this field shall take the value of 0x1FFF. Refer to the semantic definition of PCR in 2.4.3.5 and Table 2-3 for restrictions on the choice of PCR_PID value.

program_info_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the descriptors immediately following the program_info_length field.

stream_type – This is an 8-bit field specifying the type of program element carried within the packets with the PID whose value is specified by the elementary_PID. The values of stream_type are specified in Table 2-34.

NOTE – A Rec. ITU-T H.222.0 | ISO/IEC 13818-1 auxiliary stream is available for data types defined by this Specification, other than audio, video, and DSM-CC, such as the program stream directory and the program stream map.

Table 2-34 – Stream type assignments

Value	Description
0x00	ITU-T | ISO/IEC Reserved
0x01	ISO/IEC 11172-2 Video
0x02	Rec. ITU-T H.262 | ISO/IEC 13818-2 Video or ISO/IEC 11172-2 constrained parameter video stream (see Note 2)
0x03	ISO/IEC 11172-3 Audio
0x04	ISO/IEC 13818-3 Audio
0x05	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 private_sections
0x06	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 PES packets containing private data
0x07	ISO/IEC 13522 MHEG
0x08	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Annex A DSM-CC
0x09	Rec. ITU-T H.222.1
0x0A	ISO/IEC 13818-6 type A
0x0B	ISO/IEC 13818-6 type B
0x0C	ISO/IEC 13818-6 type C
0x0D	ISO/IEC 13818-6 type D
0x0E	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 auxiliary
0x0F	ISO/IEC 13818-7 Audio with ADTS transport syntax
0x10	ISO/IEC 14496-2 Visual
0x11	ISO/IEC 14496-3 Audio with the LATM transport syntax as defined in ISO/IEC 14496-3
0x12	ISO/IEC 14496-1 SL-packetized stream or FlexMux stream carried in PES packets
0x13	ISO/IEC 14496-1 SL-packetized stream or FlexMux stream carried in ISO/IEC 14496_sections

Value	Description
0x14	ISO/IEC 13818-6 Synchronized Download Protocol
0x15	Metadata carried in PES packets
0x16	Metadata carried in metadata_sections
0x17	Metadata carried in ISO/IEC 13818-6 Data Carousel
0x18	Metadata carried in ISO/IEC 13818-6 Object Carousel
0x19	Metadata carried in ISO/IEC 13818-6 Synchronized Download Protocol
0x1A	IPMP stream (defined in ISO/IEC 13818-11, MPEG-2 IPMP)
0x1B	AVC video stream conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 or AVC video sub-bitstream of SVC as defined in 2.1.10 or MVC base view sub-bitstream, as defined in 2.1.83, or AVC video sub-bitstream of MVC, as defined in 2.1.8 or MVCD base view sub-bitstream, as defined in 2.1.88, or AVC video sub-bitstream of MVCD, as defined in 2.1.9, or AVC base layer of an HEVC video stream conforming to one or more profiles defined in Annex G or Annex H of Rec. ITU-T H.265 | ISO/IEC 23008-2
0x1C	ISO/IEC 14496-3 Audio, without using any additional transport syntax, such as DST, ALS and SLS
0x1D	ISO/IEC 14496-17 Text
0x1E	Auxiliary video stream as defined in ISO/IEC 23002-3
0x1F	SVC video sub-bitstream of an AVC video stream conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10
0x20	MVC video sub-bitstream of an AVC video stream conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10
0x21	Video stream conforming to one or more profiles as defined in Rec. ITU-T T.800 | ISO/IEC 15444-1
0x22	Additional view Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream for service-compatible stereoscopic 3D services (see Notes 3 and 4)
0x23	Additional view Rec. ITU-T H.264 | ISO/IEC 14496-10 video stream conforming to one or more profiles defined in Annex A for service-compatible stereoscopic 3D services (see Notes 3 and 4)
0x24	Rec. ITU-T H.265 | ISO/IEC 23008-2 video stream or an HEVC temporal video sub-bitstream
0x25	HEVC temporal video subset of an HEVC video stream conforming to one or more profiles defined in Annex A of Rec. ITU-T H.265 | ISO/IEC 23008-2
0x26	MVCD video sub-bitstream of an AVC video stream conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10
0x27	Timeline and External Media Information Stream (see Annex T)
0x28	HEVC enhancement sub-partition which includes TemporalId 0 of an HEVC video stream where all NALs units contained in the stream conform to one or more profiles defined in Annex G of Rec. ITU-T H.265 | ISO/IEC 23008-2
0x29	HEVC temporal enhancement sub-partition of an HEVC video stream where all NAL units contained in the stream conform to one or more profiles defined in Annex G of Rec. ITU-T H.265 | ISO/IEC 23008-2
0x2A	HEVC enhancement sub-partition which includes TemporalId 0 of an HEVC video stream where all NAL units contained in the stream conform to one or more profiles defined in Annex H of Rec. ITU-T H.265 | ISO/IEC 23008-2
0x2B	HEVC temporal enhancement sub-partition of an HEVC video stream where all NAL units contained in the stream conform to one or more profiles defined in Annex H of Rec. ITU-T H.265 | ISO/IEC 23008-2
0x2C	Green access units carried in MPEG-2 sections
0x2D	ISO/IEC 23008-3 Audio with MHAS transport syntax – main stream
0x2E	ISO/IEC 23008-3 Audio with MHAS transport syntax – auxiliary stream
0x2F	Quality access units carried in sections
0x30 .. 0x7E	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 reserved
0x7F	IPMP stream
0x80 .. 0xFF	User Private

NOTE 1 – In the above table various stream types are assigned for carriage of audio signals, with or without a transport syntax. Typically, the transport syntax is used for providing sync words. The use of a specific transport syntax, if at all, is specified in the clauses in this Specification specifying the transport of the various audio signals.

NOTE 2 – Rec. ITU-T H.262 | ISO/IEC 13818-2 video with frame packing arrangement information is signalled using stream_type value 0x02.

NOTE 3 – The base view of service-compatible stereoscopic 3D services is signalled using stream_type value 0x02 for Rec. ITU-T H.262 | ISO/IEC 13818-2 video or stream_type 0x1B for Rec. ITU-T H.264 | ISO/IEC 14496-10 video.

NOTE 4 – The additional view for service-compatible stereoscopic 3D services is signalled using stream_type value 0x22 for Rec. ITU-T H.262 | ISO/IEC 13818-2 video or stream_type value 0x23 for Rec. ITU-T H.264 | ISO/IEC 14496-10 video conforming to one or more profiles defined in Annex A. For service-compatible stereoscopic 3D services, the additional view is not signalled using stream_type values 0x02 or 0x1B.

elementary_PID – This is a 13-bit field specifying the PID of the transport stream packets which carry the associated program element.

ES_info_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the descriptors of the associated program element immediately following the ES_info_length field.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex B after processing the entire transport stream program map section.

2.4.4.10 Syntax of the private section

When private data is sent in transport stream packets with a PID value designated as a program map PID in the program association table the private_section shall be used. The private_section allows data to be transmitted with a minimum of structure while enabling a decoder to parse the stream. The sections may be used in two ways: if the section_syntax_indicator is set to '1', then the whole structure common to all tables shall be used; if the indicator is set to '0', then only the fields 'table_id' through 'private_section_length' shall follow the common structure syntax and semantics and the rest of the private_section may take any form the user determines. Examples of extended use of this syntax are found in informative Annex C.

A private table may be made of several private_sections, all with the same table_id (see Table 2-35).

Table 2-35 – Private section

Syntax	No. of bits	Mnemonic
private_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
private_indicator	**1**	**bslbf**
reserved	**2**	**bslbf**
private_section_length	**12**	**uimsbf**
if (section_syntax_indicator = = '0') {		
for (i = 0; i < N; i++) {		
private_data_byte	**8**	**bslbf**
}		
}		
else {		
table_id_extension	**16**	**uimsbf**
reserved	**2**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
for (i = 0; i < private_section_length-9; i++) {		
private_data_byte	**8**	**bslbf**
}		
CRC_32	**32**	**rpchof**
}		
}		

2.4.4.11 Semantic definition of fields in private section

table_id – This 8-bit field, the value of which identifies the Private Table this section belongs to. Only values defined in Table 2-31 as "user private" may be used.

section_syntax_indicator – This is a 1-bit indicator. When set to '1', it indicates that the private section follows the generic section syntax beyond the private_section_length field. When set to '0', it indicates that the private_data_bytes immediately follow the private_section_length field.

private_indicator – This is a 1-bit user-definable flag that shall not be specified by ITU-T | ISO/IEC in the future.

private_section_length – A 12-bit field. It specifies the number of remaining bytes in the private section immediately following the private_section_length field up to the end of the private_section. The value in this field shall not exceed 4093 (0xFFD).

private_data_byte – The private_data_byte field is user definable and shall not be specified by ITU-T | ISO/IEC in the future.

table_id_extension – This is a 16-bit field. Its use and value are defined by the user.

version_number – This 5-bit field is the version number of the private_section. The version_number shall be incremented by 1 modulo 32 when a change in the information carried within the private_section occurs. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable private_section with the same table_id and section_number.

current_next_indicator – A 1-bit field, which when set to '1' indicates that the private_section sent is currently applicable. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable private_section. When the bit is set to '0', it indicates that the private_section sent is not yet applicable and shall be the next private_section with the same section_number and table_id to become valid.

section_number – This 8-bit field gives the number of the private_section. The section_number of the first section in a private table shall be 0x00. The section_number shall be incremented by 1 with each additional section in this private table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the private table of which this section is a part.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire private section.

2.4.4.12 Syntax of the transport stream section

Rec. ITU-T H.222.0 | ISO/IEC 13818-1 compliant bitstreams may carry the information defined in Table 2-36. Rec. ITU-T H.222.0 | ISO/IEC 13818-1 compliant decoders may decode the information defined in this table.

The transport stream description table is defined to support the carriage of descriptors as found in 2.6 for an entire transport stream. The descriptors shall apply to the entire transport stream. This table uses a table_id value of 0x03 as specified in Table 2-31 and is carried in transport stream packets whose PID value is 0x0002 as specified in Table 2-3.

Table 2-36 – The transport stream description table

Syntax	No. of bits	Mnemonic
TS_description_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
'0'	**1**	**bslbf**
reserved	**2**	**bslbf**
section_length	**12**	**uimsbf**
reserved	**18**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
for (i = 0; i < N; i++) {		
descriptor()		
}		
CRC_32	**32**	**rpchof**
}		

2.4.4.13 Semantic definition of fields in the transport stream section

table_id – This is an 8-bit field, which shall be set to '0x03' as specified in Table 2-31.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the section, starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

version_number – This 5-bit field is the version number of the whole transport stream description table. The version number shall be incremented by 1 modulo 32 whenever the definition of the transport stream description table changes. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable transport stream description table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable transport stream description table.

current_next_indicator – A 1-bit indicator, which, when set to '1', indicates that the transport stream description table sent is currently applicable. When the bit is set to '0', it indicates that the table sent is not yet applicable and shall be the next table to become valid.

section_number – This 8-bit field gives the number of this section. The section_number of the first section in the transport stream description table shall be 0x00. It shall be incremented by 1 with each additional section in the transport stream description table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the complete transport stream description table.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire transport stream description section.

2.5 Program stream bitstream requirements

2.5.1 Program stream coding structure and parameters

The Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program stream coding layer allows one program of one or more elementary streams to be combined into a single stream. Data from each elementary stream are multiplexed together with information that allows synchronized presentation of the elementary streams within the program.

A program stream consists of one or more elementary streams from one program multiplexed together. Audio and video elementary streams consist of access units.

Elementary stream data is carried in PES packets. A PES packet consists of a PES packet header followed by packet data. PES packets are inserted into program stream packs.

The PES packet header begins with a 32-bit start-code that also identifies the stream (refer to Table 2-22) to which the packet data belongs. The PES packet header may contain just a presentation time stamp (PTS) or both a presentation timestamp and a decoding time stamp (DTS). The PES packet header also contains other optional fields. The packet data contains a variable number of contiguous bytes from one elementary stream.

In a program stream, PES packets are organized in packs. A pack commences with a pack header and is followed by zero or more PES packets. The pack header begins with a 32-bit start-code. The pack header is used to store timing and bitrate information.

The program stream begins with a system header that optionally may be repeated. The system header carries a summary of the system parameters defined in the stream.

This Recommendation | International Standard does not specify the coded data which may be used as part of conditional access systems. This Recommendation | International Standard does, however, provide mechanisms for program service providers to transport and identify this data for decoder processing, and to correctly reference data which are here specified.

2.5.2 Program stream system target decoder

The semantics of the program stream and the constraints on these semantics require exact definitions of decoding events and the times at which these events occur. The definitions needed are set out in this Specification using a hypothetical decoder known as the program stream system target decoder (P-STD).

The P-STD is a conceptual model used to define these terms precisely and to model the decoding process during the construction of program streams. The P-STD is defined only for this purpose. Neither the architecture of the P-STD nor the timing described precludes uninterrupted, synchronized playback of program streams from a variety of decoders with different architectures or timing schedules.

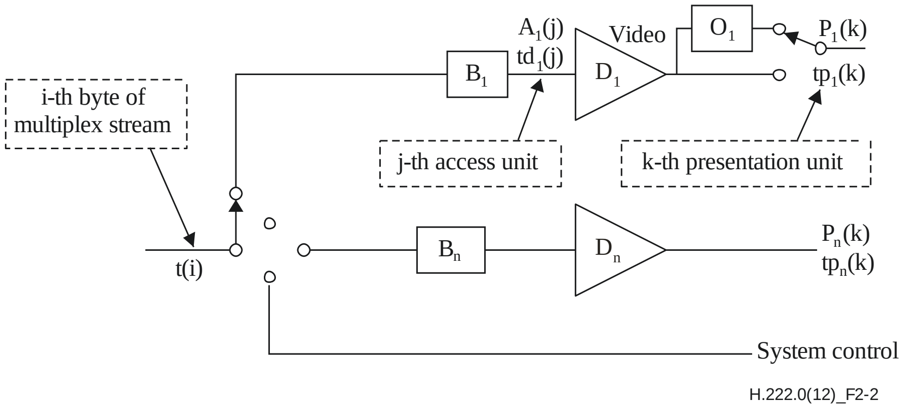


Figure 2-2 – Program stream system target decoder notation

The following notation is used to describe the program stream system target decoder and is partially illustrated in Figure 2-2.

i, i′ are indices to bytes in the program stream. The first byte has index 0.

j is an index to access units in the elementary streams.

k, k′, k″ are indices to presentation units in the elementary streams.

n is an index to the elementary streams.

t(i) indicates the time in seconds at which the i-th byte of the program stream enters the system target decoder. The value t(0) is an arbitrary constant.

SCR(i) is the time encoded in the SCR field measured in units of the 27 MHz system clock where i is the byte index of the final byte of the system_clock_reference_base field.

$A_n(j)$ is the j-th access unit in elementary stream n. $A_n(j)$ is indexed in decoding order.

$td_n(j)$ is the decoding time, measured in seconds, in the system target decoder of the j-th access unit in elementary stream n.

$P_n(k)$ is the k-th presentation unit in elementary stream n. $P_n(k)$ is indexed in presentation order.

$tp_n(k)$ is the presentation time, measured in seconds, in the system target decoder of the k-th presentation unit in elementary stream n.

t is time measured in seconds.

$F_n(t)$ is the fullness, measured in bytes, of the system target decoder input buffer for elementary stream n at time t.

B_n the input buffer in the system target decoder for elementary stream n.

BS_n is the size of the system target decoder input buffer, measured in bytes, for elementary stream n.

D_n is the decoder for elementary stream n.

O_n is the reorder buffer for video elementary stream n.

2.5.2.1 System clock frequency

Timing information referenced in P-STD is carried by several data fields defined in this Specification. The fields are defined in 2.5.3.3 and 2.4.3.6. This information is coded as the sampled value of a system clock.

The value of the system clock frequency is measured in Hz and shall meet the following constraints:

– 27 000 000 – 810 <= system_clock_frequency <= 27 000 000 + 810;

– rate of change of system_clock_frequency with time <= 75 × 10^{-3} Hz/s.

The notation "system_clock_frequency" is used in several places in this Recommendation | International Standard to refer to the frequency of a clock meeting these requirements. For notational convenience, equations in which SCR, PTS, or DTS appear, lead to values of time which are accurate to some integral multiple of (300 × 2^{33}/system_clock_frequency) seconds. This is due to the encoding of SCR timing information as 33 bits of 1/300 of the system clock frequency plus 9 bits for the remainder, and encoding as 33 bits of the system clock frequency divided by 300 for PTS and DTS.

2.5.2.2 Input to the program stream system target decoder

Data from the program stream enters the system target decoder. The i-th byte enters at time t(i). The time at which this byte enters the system target decoder can be recovered from the input stream by decoding the input system clock reference (SCR) fields and the program_mux_rate field encoded in the pack header. The SCR, as defined in equation 2-18, is coded in two parts: one, in units the period of 1/300 × the system clock frequency, called system_clock_reference_base (see equation 2-19), and one, called system_clock_reference_ext equation (see equation 2-20), in units of the period of the system clock frequency. In the following the values encoded in these fields are denoted by SCR_base(i) and SCR_ext(i). The value encoded in the SCR field indicates time t(i), where i refers to the byte containing the last bit of the system_clock_reference_base field.

Specifically:

$$SCR(i) = SCR\_base(i) \times 300 + SCR\_ext(i) \quad (2\text{-}18)$$

where:

$$SCR\_base(i) = ((system\_clock\_frequency \times t(i)) DIV\ 300) \% 2^{33} \quad (2\text{-}19)$$

$$SCR\_ext(i) = ((system\_clock\_frequency \times t(i)) DIV\ 1) \% 300 \quad (2\text{-}20)$$

The input arrival time, t(i), as given in equation 2-21, for all other bytes shall be constructed from SCR(i) and the rate at which data arrives, where the arrival rate within each pack is the value represented in the program_mux_rate field in that pack's header.

$$t(i) = \frac{SCR(i')}{system\_clock\_frequency} + \frac{i - i'}{program\_mux\_rate \times 50} \qquad (2\text{-}21)$$

where:

i′	is the index of the byte containing the last bit of the system_clock_reference_base field in the pack header
i	is the index of any byte in the pack, including the pack header
SCR(i′)	is the time encoded in the system clock reference base and extension fields in units of the system clock
program_mux_rate	is a field defined in 2.5.3.3.

After delivery of the last byte of a pack there may be a time interval during which no bytes are delivered to the input of the P-STD.

2.5.2.3 Buffering

The PES packet data from elementary stream n is passed to the input buffer for stream n, B_n. Transfer of byte i from the system target decoder input to B_n is instantaneous, so that byte i enters the buffer for stream n, of size BS_n, at time t(i).

Bytes present in the pack header, system headers, program stream maps, program stream directories, or PES packet headers of the program stream such as SCR, DTS, PTS, and packet_length fields, are not delivered to any of the buffers, but may be used to control the system.

The input buffer sizes BS_1 through BS_n are given by the P-STD buffer size parameter in the syntax in equations 2-16 and 2-17.

At the decoding time, $td_n(j)$, all data for the access unit that has been in the buffer longest, $A_n(j)$, and any stuffing bytes that immediately precede it that are present in the buffer at the time $td_n(j)$, are removed instantaneously at time $td_n(j)$. The decoding time $td_n(j)$ is specified in the DTS or PTS fields. Decoding times $td_n(j + 1)$, $td_n(j + 2)$, ... of access units without encoded DTS or PTS fields which directly follow access unit j may be derived from information in the elementary stream. Refer to Annex C of Rec. ITU-T H.262 | ISO/IEC 13818-2, ISO/IEC 13818-3, ISO/IEC 11172-2 or ISO/IEC 11172-3. Also refer to 2.7.5. As the access unit is removed from the buffer, it is instantaneously decoded to a presentation unit.

The program stream shall be constructed and t(i) shall be chosen so that the input buffers of size BS_1 through BS_n neither overflow nor underflow in the program system target decoder. That is:

$$0 \le F_n(t) \le BS_n$$

for all t and n,

and:

$$F_n(t) = 0$$

instantaneously before t = t(0).

$F_n(t)$ is the instantaneous fullness of P-STD buffer B_n.

An exception to this condition is that the P-STD buffer B_n may underflow when the low_delay flag in the video sequence header is set to '1' (refer to 2.4.2.7) or when trick_mode status is true (refer to 2.4.3.8).

For all program streams, the delay caused by system target decoder input buffering shall be less than or equal to one second except for still picture video data and ISO/IEC 14496 streams. The input buffering delay is the difference in time between a byte entering the input buffer and when it is decoded.

Specifically: in the case of no still picture video data and no ISO/IEC 14496 stream the delay is constrained by:

$$tdn(j) - t(i) <= 1\,\text{s}$$

in the case of still picture video data the delay is constrained by:

$$tdn(j) - t(i) <= 60 \text{ s}$$

in the case of ISO/IEC 14496 streams the delay is constrained by:

$$tdn(j) - t(i) <= 10 \text{ s}$$

for all bytes contained in access unit j.

For program streams, all bytes of each pack shall enter the P-STD before any byte of a subsequent pack.

When the **low_delay** flag in the video sequence extension is set to '1' (refer to 6.2.2.3 of Rec. ITU-T H.262 | ISO/IEC 13818-2), the VBV buffer may underflow. In this case when the P-STD elementary stream buffer B_n is examined at the time specified by $td_n(j)$, the complete data for the access unit may not be present in the buffer B_n. When this case arises, the buffer shall be re-examined at intervals of two field-periods until the data for the complete access unit is present in the buffer. At this time the entire access unit shall be removed from buffer B_n instantaneously.

VBV buffer underflow is allowed to occur continuously without limit. The P-STD decoder shall remove access unit data from buffer B_n at the earliest time consistent with the paragraph above and any DTS or PTS values encoded in the bitstream. The decoder may be unable to re-establish correct decoding and display times as indicated by DTS and PTS until the VBV buffer underflow situation ceases and a PTS or DTS is found in the bitstream.

2.5.2.4 PES streams

It is possible to construct a stream of data as a contiguous stream of PES packets each containing data of the same elementary stream and with the same stream_id. Such a stream is called a PES stream. The PES-STD model for a PES stream is identical to that for the program stream, with the exception that the Elementary Stream Clock Reference (ESCR) is used in place of the SCR, and ES_rate in place of program_mux_rate. The demultiplexor sends data to only one elementary stream buffer.

NOTE – In the following equations, unit conversion should be implicitly performed as appropriate.

As a PES stream only carries a single elementary stream, the buffer sizes in the PES-STD do not account for multiplexing with other elementary streams, but only for multiplexing of the elementary stream carried in the PES stream with PES headers, pack headers and system headers.

Buffer sizes BS_n in the PES-STD model are defined as follows:

– *For Rec. ITU-T H.262 | ISO/IEC 13818-2 video:*

$$BS_n = VBV_{max}[\text{profile, level}] + BS_{oh}$$

BS_{oh} = (1/750) seconds × R_{max}[profile, level], where VBV_{max}[profile, level] and R_{max}[profile, level] are the maximum VBV size and bit rate per profile, level, and layer as defined in Tables 8-14 and 8-13, respectively, of Rec. ITU-T H.262 | ISO/IEC 13818-2. BS_{oh} is allocated for PES packet header overhead.

– *For ISO/IEC 11172-2 video:*

$$BS_n = VBV_{max} + BS_{oh}$$

BS_{oh} = (1/750) seconds × R_{max}, where R_{max} and vbv_max refer to the maximum bitrate and maximum vbv_buffer_size for a constrained parameter bitstream in ISO/IEC 11172-2 respectively.

– *For ISO/IEC 11172-3 or ISO/IEC 13818-3 audio:*

$$BS_n = 2848 \text{ bytes}$$

– *For ISO/IEC 13818-7 ADTS audio*:

$$BS_n = 2848 \text{ bytes if 1-2 channels}$$
$$BS_n = 7200 \text{ bytes if 3-8 channels}$$
$$BS_n = 10800 \text{ bytes if 9-12 channels}$$
$$BS_n = 43200 \text{ bytes if 13-48 channels}$$

Note that the above numbers differ from the BS_n numbers specified in 2.4.3.2 due to the fact that a PES stream carries a single elementary stream only.

– *For ISO/IEC 14496-3 audio, except for ISO/IEC 14496-3 DST, ALS and SLS*:

$$BS_n = 2848 \text{ bytes if 1-2 channels}$$
$$BS_n = 7200 \text{ bytes if 3-8 channels}$$
$$BS_n = 10800 \text{ bytes if 9-12 channels}$$
$$BS_n = 43200 \text{ bytes if 13-48 channels}$$

Note that the above numbers differ from the BSn numbers specified in 2.11.2.2 due to the fact that a PES stream carries a single elementary stream only.

– *For ISO/IEC 14496-3 DST-64 audio*:

$$BS_n = 5000 \times (\text{number of channels}) \text{ bytes; hence for stereo } BS_n = 10\,000 \text{ bytes and for 5.1 surround sound audio } BS_n = 30\,000 \text{ bytes}$$

– *For ISO/IEC 14496-3 DST-128 audio*:

$$BS_n = 10\,000 \times (\text{number of channels}) \text{ bytes; hence for stereo } BS_n = 20\,000 \text{ bytes and for 5.1 surround sound audio } BS_n = 60\,000 \text{ bytes}$$

– *For ISO/IEC 14496-3 DST-256 audio*:

$$BS_n = 20\,000 \times (\text{number of channels}) \text{ bytes; hence for stereo } BS_n = 40\,000 \text{ bytes and for 5.1 surround sound audio } BS_n = 120\,000 \text{ bytes}$$

– *For ISO/IEC 14496-3 ALS and SLS audio*:

$$BS_n = 33\,000 \times (\text{number of channels}) \text{ bytes; hence for stereo } BS_n = 66\,000 \text{ bytes and for 5.1 surround sound audio } BS_n = 198\,000 \text{ bytes}$$

– *For Rec. ITU-T H.264 | ISO/IEC 14496-10 video:*

$$BS_n = 1200 \times \text{MaxCPB[level]} + BS_{oh}$$

where MaxCPB[level] is defined in Table A.1 (Level Limits) in Rec. ITU-T H.264 | ISO/IEC 14496-10 for each level.

2.5.2.5 Decoding and presentation

Decoding and presentation in the program stream system target decoder are the same as defined for the transport stream system target decoder in 2.4.2.5 and 2.4.2.6 respectively.

2.5.2.6 P-STD extensions for carriage of ISO/IEC 14496 data

For decoding of ISO/IEC 14496 data carried in a program stream the P-STD model is extended. For decoding of individual ISO/IEC 14496 elementary streams in the P-STD see 2.11.2. Clause 2.11.3 defines P-STD extensions and parameters for decoding of ISO/IEC 14496 scenes and associated streams.

2.5.2.7 P-STD extensions for carriage of Rec. ITU-T H.264 | ISO/IEC 14496-10 video

For decoding of AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 video streams carried in a program stream in the P-STD model, see 2.14.3.2, for decoding of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10 carried in a program stream in the P-STD model, see 2.14.3.6 and for decoding of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10 carried in a program stream in the P-STD model, see 2.14.3.8.

2.5.2.8 P-STD extensions for carriage of ISO/IEC 14496-17 text streams

For decoding of ISO/IEC 14496-17 text streams carried in a program stream in the P-STD model, see 2.15.3.2.

2.5.3 Specification of the program stream syntax and semantics

The following syntax describes a stream of bytes.

2.5.3.1 Program stream

See Table 2-37.

Table 2-37 – Program stream

Syntax	No. of bits	Mnemonic
MPEG2_program_stream() {		
do {		
pack()		
} while (nextbits() = = pack_start_code)		
MPEG_program_end_code	**32**	**bslbf**
}		

2.5.3.2 Semantic definition of fields in program stream

MPEG_program_end_code – The MPEG_program_end_code is the bit string '0000 0000 0000 0000 0000 0001 1011 1001' (0x000001B9). It terminates the program stream.

2.5.3.3 Pack layer of program stream

See Tables 2-38 and 2-39.

Table 2-38 – Program stream pack

Syntax	No. of bits	Mnemonic
pack() {		
pack_header()		
while (nextbits() = -= packet_start_code_prefix) {		
PES_packet()		
}		
}		

Table 2-39 – Program stream pack header

Syntax	No. of bits	Mnemonic
pack_header() {		
pack_start_code	**32**	**bslbf**
'01'	**2**	**bslbf**
system_clock_reference_base [32..30]	**3**	**bslbf**
marker_bit	**1**	**bslbf**
system_clock_reference_base [29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
system_clock_reference_base [14..0]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
system_clock_reference_extension	**9**	**uimsbf**
marker_bit	**1**	**bslbf**
program_mux_rate	**22**	**uimsbf**
marker_bit	**1**	**bslbf**
marker_bit	**1**	**bslbf**
reserved	**5**	**bslbf**
pack_stuffing_length	**3**	**uimsbf**
for (i = 0; i < pack_stuffing_length; i++) {		
stuffing_byte	**8**	**bslbf**
}		
if (nextbits() = = system_header_start_code) {		
system_header ()		
}		
}		

2.5.3.4 Semantic definition of fields in program stream pack

pack_start_code – The pack_start_code is the bit string '0000 0000 0000 0000 0000 0001 1011 1010' (0x000001BA). It identifies the beginning of a pack.

system_clock_reference_base; system_clock_reference_extension – The system clock reference (SCR) is a 42-bit field coded in two parts. The first part, system_clock_reference_base, is a 33-bit field whose value is given by SCR_base(i) as given in equation 2-19. The second part, system_clock_reference_extension, is a 9-bit field whose value is given by SCR_ext(i), as given in equation 2-20. The SCR indicates the intended time of arrival of the byte containing the last bit of the system_clock_reference_base at the input of the program target decoder.

The frequency of coding requirements for the SCR field are given in 2.7.1.

marker_bit – A marker_bit is a 1-bit field that has the value '1'.

program_mux_rate – This is a 22-bit integer specifying the rate at which the P-STD receives the program stream during the pack in which it is included. The value of program_mux_rate is measured in units of 50 bytes/second. The value '0' is forbidden. The value represented in program_mux_rate is used to define the time of arrival of bytes at the input to the P-STD in 2.5.2. The value encoded in the program_mux_rate field may vary from pack to pack in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program multiplexed stream.

pack_stuffing_length – A 3-bit integer specifying the number of stuffing bytes which follow this field.

stuffing_byte – This is a fixed 8-bit value equal to '1111 1111' that can be inserted by the encoder, for example to meet the requirements of the channel. It is discarded by the decoder. In each pack header no more than 7 stuffing bytes shall be present.

2.5.3.5 System header

See Table 2-40.

Table 2-40 – Program stream system header

Syntax	No. of bits	Mnemonic
system_header () {		
system_header_start_code	**32**	**bslbf**
header_length	**16**	**uimsbf**
marker_bit	**1**	**bslbf**
rate_bound	**22**	**uimsbf**
marker_bit	**1**	**bslbf**
audio_bound	**6**	**uimsbf**
fixed_flag	**1**	**bslbf**
CSPS_flag	**1**	**bslbf**
system_audio_lock_flag	**1**	**bslbf**
system_video_lock_flag	**1**	**bslbf**
marker_bit	**1**	**bslbf**
video_bound	**5**	**uimsbf**
packet_rate_restriction_flag	**1**	**bslbf**
reserved_bits	**7**	**bslbf**
while (nextbits () == '1') {		
stream_id	**8**	**uimsbf**
if (stream_id == '1011 0111') {		
'11'	**2**	**bslbf**
'000 0000'	**7**	**bslbf**
stream_id_extension	**7**	**uimsbf**
'1011 0110'	**8**	**bslbf**
'11'	**2**	**bslbf**
P-STD_buffer_bound_scale	**1**	**bslbf**
P-STD_buffer_size_bound	**13**	**uimsbf**
}		
else {		
'11'	**2**	**bslbf**
P-STD_buffer_bound_scale	**1**	**bslbf**
P-STD_buffer_size_bound	**13**	**uimsbf**
}		
}		
}		

2.5.3.6 Semantic definition of fields in system header

system_header_start_code – The system_header_start_code is the bit string '0000 0000 0000 0000 0000 0001 1011 1011' (0x000001BB). It identifies the beginning of a system header.

header_length – This 16-bit field indicates the length in bytes of the system header following the header_length field. Future extensions of this Specification may extend the system header.

rate_bound – A 22-bit field. The rate_bound is an integer value greater than or equal to the maximum value of the program_mux_rate field coded in any pack of the program stream. It may be used by a decoder to assess whether it is capable of decoding the entire stream.

audio_bound – A 6-bit field. The audio_bound is an integer in the inclusive range from 0 to 32 and is set to a value greater than or equal to the maximum number of ISO/IEC 13818-3 and ISO/IEC 11172-3 audio streams in the program stream for which the decoding processes are simultaneously active. For the purpose of this subclause, the decoding process of an ISO/IEC 13818-3 or ISO/IEC 11172-3 audio stream is active if the STD buffer is not empty or if a Presentation Unit is being presented in the P-STD model.

fixed_flag – The fixed_flag is a 1-bit flag. When set to '1' fixed bitrate operation is indicated. When set to '0' variable bitrate operation is indicated. During fixed bitrate operation, the value encoded in all system_clock_reference fields in the multiplexed Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream shall adhere to the following linear equation:

$$SCR\_base(i) = ((\text{c1} \times \text{i} + \text{c2})\ DIV\ 300)\ \%\ 2^{33} \quad (2\text{-}22)$$

$$SCR\_ext(i) = ((\text{c1} \times \text{i} + \text{c2})\ DIV\ 300)\ \%\ 300 \quad (2\text{-}23)$$

where:

c1 is a real-valued constant valid for all i.

c2 is a real-valued constant valid for all i.

i is the index in the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 multiplexed stream of the byte containing the final bit of any system_clock_reference field in the stream.

CSPS_flag – The CSPS_flag is a 1-bit field. If its value is set to '1' the program stream meets the constraints defined in 2.7.9.

system_audio_lock_flag – The system_audio_lock_flag is a 1-bit field indicating that there is a specified, constant rational relationship between the audio sampling rate and the system_clock_frequency in the system target decoder. The system_clock_frequency is defined in 2.5.2.1 and the audio sampling rate is specified in ISO/IEC 13818-3. The system_audio_lock_flag may only be set to '1' if, for all presentation units in all audio elementary streams in the program stream, the ratio of system_clock_frequency to the actual audio sampling rate, SCASR, is constant and equal to the value indicated in the following table at the nominal sampling rate indicated in the audio stream.

$$SCASR = \frac{system\_clock\_frequency}{audio\_sample\_rate\_in\_the\_P-STD} \quad (2\text{-}24)$$

The notation $\frac{X}{Y}$ denotes real division.

Nominal audio sampling frequency (kHz)	16	32	22.05	44.1	24	48
SCASR	27 000 000 ------------- 16 000	27 000 000 ------------- 32 000	27 000 000 ------------- 22 050	27 000 000 ------------- 44 100	27 000 000 ------------- 24 000	27 000 000 ------------- 48 000

system_video_lock_flag – The system_video_lock_flag is a 1-bit field indicating that there is a specified, constant rational relationship between the video time base and the system clock frequency in the system target decoder. The system_video_lock_flag may only be set to '1' if, for all presentation units in all video elementary streams in the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, the ratio of system_clock_frequency to the frequency of the actual video time base is constant.

For ISO/IEC 11172-2 and Rec. ITU-T H.262 | ISO/IEC 13818-2 video streams, if the system_video_lock_flag is set to '1', then the ratio of system_clock_frequency to the actual video frame rate, SCFR, shall be constant and equal to the value indicated in the following table at the nominal frame rate indicated in the video stream.

For ISO/IEC 14496-2 video streams, if the system_video_lock_flag is set to '1', then the time base of the ISO/IEC 14496-2 video stream, as defined by vop_time_increment_resolution, shall be locked to the STC and shall be exactly equal to N times system_clock_frequency divided by K, with N and K integers that have a fixed value within each visual object sequence, with K greater than or equal to N.

For Rec. ITU-T H.264 | ISO/IEC 14496-10 video streams, the frequency of the AVC time base is defined by the AVC parameter time_scale. If the system_video_lock_flag is set to '1' for an AVC video stream or for a video sub-bitstream, then the frequency of the AVC time base shall be locked to the STC and shall be exactly equal to N times system_clock_frequency divided by K, with N and K integers that have a fixed value within each AVC video sequence, with K greater than or equal to N.

$$SCFR = \frac{system\_clock\_frequency}{frame\_rate\_in\_the\_P-STD} \quad (2\text{-}25)$$

Nominal frame rate (Hz)	23.976	24	25	29.97	30	50	59.94	60
SCFR	1 126 125	1 125 000	1 080 000	900 900	900 000	540 000	450 450	450 000

The values of the ratio SCFR are exact. The actual frame rate differs slightly from the nominal rate in cases where the nominal rate is 23.976, 29.97, or 59.94 frames per second.

video_bound – The video_bound is a 5-bit integer in the inclusive range from 0 to 16 and is set to a value greater than or equal to the maximum number of video streams in the program stream of which the decoding processes are simultaneously active. For the purpose of this subclause, the decoding process of a video stream is active if one of the buffers in the P-STD model is not empty, or if a Presentation Unit is being presented in the P-STD model.

packet_rate_restriction_flag – The packet_rate_restriction_flag is a 1-bit flag. If the CSPS flag is set to '1', the packet_rate_restriction_flag indicates which constraint is applicable to the packet rate, as specified in 2.7.9. If the CSPS flag is set to value of '0', then the meaning of the packet_rate_restriction_flag is undefined.

reserved_bits – This 7-bit field is reserved for future use by ISO/IEC. Until otherwise specified by ITU-T | ISO/IEC it shall have the value '111 1111'.

stream_id – The stream_id is an 8-bit field that indicates the coding and elementary stream number of the stream to which the following P-STD_buffer_bound_scale and P-STD_buffer_size_bound fields refer.

If stream_id equals '1011 1000' the P-STD_buffer_bound_scale and P-STD_buffer_size_bound fields following the stream_id refer to all audio streams in the program stream.

If stream_id equals '1011 1001' the P-STD_buffer_bound_scale and P-STD_buffer_size_bound fields following the stream_id refer to all video streams in the program stream.

If stream_id equals '1111 1101', the P-STD_buffer_bound_scale and P-STD_buffer_size_bound fields following the stream_id refer to all elementary streams with an extended_stream_id in the program stream, independent of the coded value of the stream_id_extension in the PES header of those streams.

If stream_id equals '1011 0111', the following stream_id_extension field shall be interpreted as referring to the stream coding and elementary stream number according to Table 2-27.

If the stream_id takes on any other value it shall be a byte value greater than or equal to '1011 1100' and shall be interpreted as referring to the stream coding and elementary stream number according to Table 2-22.

Each elementary stream present in the program stream shall have its P-STD_buffer_bound_scale and P-STD_buffer_size_bound specified exactly once by this mechanism in each system header.

stream_id_extension – The stream_id_extension is a 7-bit field. In case the stream_id field is coded with the value '1011 0111', then the stream_id_extension indicates the coding and elementary stream number of the stream with an extended_stream_id to which the P-STD_buffer_bound_scale and P-STD_buffer_size_bound fields following the stream_id_extension field refer.

P-STD_buffer_bound_scale – The P-STD_buffer_bound_scale is a 1-bit field that indicates the scaling factor used to interpret the subsequent P-STD_buffer_size_bound field. If the preceding stream_id indicates an audio stream, P-STD_buffer_bound_scale shall have the value '0'. If the preceding stream_id indicates a video stream, P-STD_buffer_bound_scale shall have the value '1'. For all other stream types, the value of the P-STD_buffer_bound_scale may be either '1' or '0'.

P-STD_buffer_size_bound – The P-STD_buffer_size_bound is a 13-bit unsigned integer defining a value greater than or equal to the maximum P-STD input buffer size, BS_n, over all packets for stream n in the program stream. If P-STD_buffer_bound_scale has the value '0', then P-STD_buffer_size_bound measures the buffer size bound in units of 128 bytes. If P-STD_buffer_bound_scale has the value '1', then P-STD_buffer_size_bound measures the buffer size bound in units of 1024 bytes. Thus:

$$if\ (P-STD\_buffer\_bound\_scale == 0)$$
$$BS_n \leq P-STD\_buffer\_size\_bound \times 128$$

else:

$$BS_n \leq P-STD\_buffer\_size\_bound \times 1024$$

2.5.3.7 Packet layer of program stream

The packet layer of the program stream is defined by the PES packet layer in 2.4.3.6.

2.5.4 Program stream map

The Program Stream Map (PSM) provides a description of the elementary streams in the program stream and their relationship to one another. When carried in a transport stream this structure shall not be modified. The PSM is present as a PES packet when the stream_id value is 0xBC (refer to Table 2-22).

NOTE – This syntax differs from the PES packet syntax described in 2.4.3.6.

Definition for the descriptor() fields may be found in 2.6.

2.5.4.1 Syntax of program stream map

See Table 2-41.

Table 2-41 – Program stream map

Syntax	No. of bits	Mnemonic
program_stream_map() {		
packet_start_code_prefix	**24**	**bslbf**
map_stream_id	**8**	**uimsbf**
program_stream_map_length	**16**	**uimsbf**
current_next_indicator	**1**	**bslbf**
single_extension_stream_flag	**1**	**bslbf**
reserved	**1**	**bslbf**
program_stream_map_version	**5**	**uimsbf**
reserved	**7**	**bslbf**
marker_bit	**1**	**bslbf**
program_stream_info_length	**16**	**uimsbf**
for (i = 0; i < N; i++) {		
descriptor()		
}		
elementary_stream_map_length	**16**	**uimsbf**
for (i = 0; i < N1; i++) {		
stream_type	**8**	**uimsbf**
elementary_stream_id	**8**	**uimsbf**
elementary_stream_info_length	**16**	**Uimsbf**
if (elementary_stream_id = = 0xFD &&		
single_extension_stream_flag = = 0) {		
pseudo_descriptor_tag	**8**	**Uimsbf**
pseudo_descriptor_length	**8**	**Uimsbf**
marker_bit	**1**	**Bslbf**
elementary_stream_id_extension	**7**	**Uimsbf**
for (i = 3; i < N2; i++) {		
descriptor()		
}		
}		
else {		
for (i = 0; i < N2; i++) {		
descriptor()		
}		
}		
}		
CRC_32	**32**	**rpchof**

Syntax	No. of bits	Mnemonic
}		

2.5.4.2 Semantic definition of fields in program stream map

packet_start_code_prefix – The packet_start_code_prefix is a 24-bit code. Together with the map_stream_id that follows it constitutes a packet start code that identifies the beginning of a packet. The packet_start_code_prefix is the bit string '0000 0000 0000 0000 0000 0001' (0x000001 in hexadecimal).

map_stream_id – This is an 8-bit field whose value shall be 0xBC.

program_stream_map_length – The program_stream_map_length is a 16-bit field indicating the total number of bytes in the program_stream_map immediately following this field. The maximum value of this field is 1018 (0x3FA).

single_extension_stream_flag – This is a 1-bit field indicating, when set to '1', that the program stream contains at most one elementary stream with stream_id equal to 0xFD.

current_next_indicator – This is a 1-bit field, when set to '1' indicates that the program stream map sent is currently applicable. When the bit is set to '0', it indicates that the program stream map sent is not yet applicable and shall be the next table to become valid.

program_stream_map_version – This 5-bit field is the version number of the whole program stream map. The version number shall be incremented by 1 modulo 32 whenever the definition of the program stream map changes. When the current_next_indicator is set to '1', then the program_stream_map_version shall be that of the currently applicable program stream map. When the current_next_indicator is set to '0', then the program_stream_map_version shall be that of the next applicable program stream map.

program_stream_info_length – The program_stream_info_length is a 16-bit field indicating the total length of the descriptors immediately following this field.

marker_bit – A marker_bit is a 1-bit field that has the value '1'.

elementary_stream_map_length – This is a 16-bit field specifying the total length, in bytes, of all elementary stream information in this program stream map. It includes the stream_type, elementary_stream_id, and elementary_stream_info_length fields.

stream_type – This 8-bit field specifies the type of the stream according to Table 2-34. The stream_type field shall only identify elementary streams contained in PES packets. A value of 0x05 is prohibited.

elementary_stream_id – The elementary_stream_id is an 8-bit field indicating the value of the stream_id field in the PES packet headers of PES packets in which this elementary stream is stored. When elementary_stream_id is equal to 0xFD, the following applies:

- If single_extension_stream_flag is equal to 1, this indicates that the program stream contains only one elementary stream with stream_id equal to 0xFD. Note that the type of this elementary stream is signalled by the encoded value of the stream_id_extension field in the PES headers of PES packets carrying this elementary stream.
- Otherwise (single_extension_stream_flag is equal to 0), the elementary_stream_id_extension field is present to identify the elementay stream.

elementary_stream_info_length – The elementary_stream_info_length is a 16-bit field indicating the length in bytes of the descriptors and, when present, the pseudo_descriptor_tag, the pseudo_descriptor_length, and the elementary_stream_id_extension (and associated marker_bit) data immediately following this field.

pseudo_descriptor_tag – This is an 8-bit unsigned integer that shall be coded with the value 0x01; note that the use of value 0x01 for descriptor tags is forbidden in Table 2-45.

pseudo_descriptor_length – The pseudo_descriptor_length is an 8-bit unsigned integer that shall be coded with the value 1.

elementary_stream_id_extension – This 7-bit field, when present, indicates the encoded value of the elementary_stream_id_extension field in the PES packet headers of PES packets in which this elementary stream is stored.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire program stream map.

2.5.5 Program stream directory

The directory for an entire stream is made up of all the directory data carried by program stream directory packets identified with the directory_stream_id. The syntax for program_stream_directory packets is defined in Table 2-42.

NOTE 1 – This syntax differs from the PES packet syntax described in 2.4.3.6.

Directory entries may be required to reference I-pictures in a video stream as defined in Rec. ITU-T H.262 | ISO/IEC 13818-2 and ISO/IEC 11172-2. If an I-picture that is referenced in a directory entry is preceded by a sequence header with no intervening picture headers, the directory entry shall reference the first byte of the sequence header. If an I-picture that is referenced in a directory entry is preceded by a group of pictures header with no intervening picture headers and no immediately preceding sequence header, the directory entry shall reference the first byte of the group of pictures header. Any other picture that a directory entry references shall be referenced by the first byte of the picture header.

NOTE 2 – It is recommended that I-pictures immediately following a sequence header should be referenced in directory structures so that the directory contains an entry at every point where the decoder may be reset completely.

For AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10, directory entries may be required to reference IDR picture or pictures associated with a recovery point SEI message in an AVC video stream. Each such directory entry shall refer to the first byte of an AVC access unit.

For video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, directory entries may be required to reference IDR picture or pictures to be re-assembled from video sub-bitstreams and associated with a recovery point SEI message present in a video sub-bitstream. Each such directory entry shall refer to the first byte of an SVC dependency representation.

For MVC video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, directory entries may be required to reference IDR picture or pictures to be re-assembled from MVC video sub-bitstreams and associated with a recovery point SEI message present in an MVC video sub-bitstream. Each such directory entry shall refer to the first byte of an MVC view-component subset.

Directory references to audio streams as defined in ISO/IEC 13818-3 and ISO/IEC 11172-3 shall be the syncword of the audio frame.

NOTE 3 – It is recommended that the distance between referenced access units not exceed half a second.

Access units shall be referenced in a program_stream_directory packet in the same order that they appear in the bitstream.

2.5.5.1 Syntax of program stream directory packet

See Table 2-42.

Table 2-42 – Program stream directory packet

Syntax	No. of bits	Mnemonic
directory_PES_packet(){		
packet_start_code_prefix	**24**	**bslbf**
directory_stream_id	**8**	**uimsbf**
PES_packet_length	**16**	**uimsbf**
number_of_access_units	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
prev_directory_offset[44..30]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
prev_directory_offset[29..15]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
prev_directory_offset[14..0]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
next_directory_offset[44..30]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
next_directory_offset[29..15]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
next_directory_offset[14..0]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
for (i = 0; i < number_of_access_units; i++) {		
packet_stream_id	**8**	**uimsbf**
PES_header_position_offset_sign	**1**	**tcimsbf**
PES_header_position_offset[43..30]	**14**	**uimsbf**
marker_bit	**1**	**bslbf**
PES_header_position_offset[29..15]	**15**	**uimsbf**

Syntax	No. of bits	Mnemonic
marker_bit	**1**	**bslbf**
PES_header_position_offset[14..0]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
reference_offset	**16**	**uimsbf**
marker_bit	**1**	**bslbf**
if (packet_stream_id = = 0xFD) {		
packet_stream_id_extension_msbs	**3**	**uimsbf**
}		
else {		
reserved	**3**	**bslbf**
}		
PTS[32..30]	**3**	**uimsbf**
marker_bit	**1**	**bslbf**
PTS[29..15]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
PTS[14..0]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
bytes_to_read[22..8]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
bytes_to_read[7..0]	**8**	**uimsbf**
marker_bit	**1**	**bslbf**
intra_coded_indicator	**1**	**bslbf**
coding_parameters_indicator	**2**	**bslbf**
if (packet_stream_id = = 0xFD) {		
packet_stream_id_extension_lsbs	**4**	**uimsbf**
}		
else {		
reserved	**4**	**bslbf**
}		
}		
}		

2.5.5.2 Semantic definition of fields in program stream directory

packet_start_code_prefix – The packet_start_code_prefix is a 24-bit code. Together with the stream_id that follows, it constitutes a packet start code that identifies the beginning of a packet. The packet_start_code_prefix is the bit string '0000 0000 0000 0000 0000 0001' (0x000001 in hexadecimal).

directory_stream_id – This 8-bit field shall have a value '1111 1111' (0xFF).

PES_packet_length – The PES_packet_length is a 16-bit field indicating the total number of bytes in the program_stream_directory immediately following this field (refer to Table 2-21).

number_of_access_units – This 15-bit field is the number of access_units that are referenced in this Directory PES packet.

prev_directory_offset – This 45-bit unsigned integer gives the byte address offset of the first byte of the packet start code of the previous program stream directory packet. This address offset is relative to the first byte of the start code of the packet which contains this previous_directory_offset field. The value '0' indicates that there is no previous program stream directory packet.

next_directory_offset – This 45-bit unsigned integer gives the byte address offset of the first byte of the packet start code of the next program stream directory packet. This address offset is relative to the first byte of the start code of the packet which contains this next_directory_offset field. The value '0' indicates that there is no next program stream directory packet.

packet_stream_id – This 8-bit field is the stream_id of the elementary stream that contains the access unit referenced by this directory entry.

PES_header_position_offset_sign – This 1-bit field is the arithmetic sign for the PES_header_position_offset described immediately following. A value of '0' indicates that the PES_header_position_offset is a positive offset. A value of '1' indicates that the PES_header_position_offset is a negative offset.

PES_header_position_offset – This 44-bit unsigned integer gives the byte offset address of the first byte of the PES packet containing the access unit referenced. The offset address is relative to the first byte of the start-code of the packet containing this PES_header_position_offset field. The value '0' indicates that no access unit is referenced.

reference_offset – This 16-bit field is an unsigned integer indicating the position of the first byte of the referenced access unit, measured in bytes relative to the first byte of the PES packet containing the first byte of the referenced access unit.

PTS (presentation_time_stamp) – This 33-bit field is the PTS of the access unit that is referenced. The semantics of the coding of the PTS field are as described in 2.4.3.6.

bytes_to_read – This 23-bit unsigned integer is the number of bytes in the program stream after the byte indicated by reference_offset that are needed to decode the access unit completely. This value includes any bytes multiplexed at the systems layer including those containing information from other streams.

intra_coded_indicator – This is a 1-bit flag. When set to '1' it indicates that the referenced access unit is not predictively coded. This is independent of other coding parameters that might be needed to decode the access unit. For example, this field shall be coded as '1' for video Intra frames, whereas for 'P' and 'B' frames this bit shall be coded as '0'. For all PES packets containing data which is not from a Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream, this field is undefined (see Table 2-43).

Table 2-43 – Intra_coded indicator

Value	Meaning
'0'	Not Intra
'1'	Intra

coding_parameters_indicator – This 2-bit field is used to indicate the location of coding parameters that are needed to decode the access units referenced (see Table 2-44). For example, this field can be used to determine the location of quantization matrices for video frames.

Table 2-44 – Coding_parameters indicator

Value	Meaning
'00'	All coding parameters are set to their default values
'01'	All coding parameters are set in this access unit, at least one of them is not set to a default
'10'	Some coding parameters are set in this access unit
'11'	No coding parameters are coded in this access unit

packet_stream_id_extension_msbs – This 3-bit field is present if packet_stream_id equals 0xFD; its coding is specified below.

packet_stream_id_extension_lsbs – This 4-bit field is present if packet_stream_id equals 0xFD; its coding is specified below.

If packet_stream_id is equal to 0xFD, the packet_stream_id_extension indicates the encoded value of the stream_id_extension in the PES header of the PES packet(s) containing the access unit referenced by this directory entry. The value of the packet_stream_id_extension is specified by:

*packet_stream_id_extension = packet_stream_id_extension_msbs * 16 + packet_stream_id_extension_lsbs*

2.6 Program and program element descriptors

Program and program element descriptors are structures which may be used to extend the definitions of programs and program elements. All descriptors have a format which begins with an 8-bit tag value. The tag value is followed by an 8-bit descriptor length and data fields.

2.6.1 Semantic definition of fields in program and program element descriptors

The following semantics apply to the descriptors defined in 2.6.2 through the end of 2.6.

descriptor_tag – The descriptor_tag is an 8-bit field which identifies each descriptor.

Table 2-45 provides the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined, Rec. ITU-T H.222.0 | ISO/IEC 13818-1 reserved, and user available descriptor tag values. An 'X' in the TS or PS columns indicates the applicability of the descriptor to either the transport stream or program stream respectively. Note that the meaning of fields in a descriptor may depend on which stream it is used in. Each case is specified in the descriptor semantics below.

descriptor_length – The descriptor_length is an 8-bit field specifying the number of bytes of the descriptor immediately following descriptor_length field.

Table 2-45 – Program and program element descriptors

descriptor_tag	TS	PS	Identification
0	n/a	n/a	Reserved
1	n/a	X	Forbidden
2	X	X	video_stream_descriptor
3	X	X	audio_stream_descriptor
4	X	X	hierarchy_descriptor
5	X	X	registration_descriptor
6	X	X	data_stream_alignment_descriptor
7	X	X	target_background_grid_descriptor
8	X	X	video_window_descriptor
9	X	X	CA_descriptor
10	X	X	ISO_639_language_descriptor
11	X	X	system_clock_descriptor
12	X	X	multiplex_buffer_utilization_descriptor
13	X	X	copyright_descriptor
14	X		maximum_bitrate_descriptor
15	X	X	private_data_indicator_descriptor
16	X	X	smoothing_buffer_descriptor
17	X		STD_descriptor
18	X	X	IBP_descriptor
19 .. 26	X		Defined in ISO/IEC 13818-6
27	X	X	MPEG-4_video_descriptor
28	X	X	MPEG-4_audio_descriptor
29	X	X	IOD_descriptor
30	X		SL_descriptor
31	X	X	FMC_descriptor
32	X	X	external_ES_ID_descriptor
33	X	X	MuxCode_descriptor
34	X	X	FmxBufferSize_descriptor
35	X		multiplexBuffer_descriptor
36	X	X	content_labeling_descriptor
37	X	X	metadata_pointer_descriptor
38	X	X	metadata_descriptor
39	X	X	metadata_STD_descriptor
40	X	X	AVC video descriptor
41	X	X	IPMP_descriptor (defined in ISO/IEC 13818-11, MPEG-2 IPMP)
42	X	X	AVC timing and HRD descriptor
43	X	X	MPEG-2_AAC_audio_descriptor
44	X	X	FlexMuxTiming_descriptor
45	X	X	MPEG-4_text_descriptor
46	X	X	MPEG-4_audio_extension_descriptor
47	X	X	Auxiliary_video_stream_descriptor
48	X	X	SVC extension descriptor
49	X	X	MVC extension descriptor
50	X	n/a	J2K video descriptor
51	X	X	MVC operation point descriptor
52	X	X	MPEG2_stereoscopic_video_format_descriptor
53	X	X	Stereoscopic_program_info_descriptor
54	X	X	Stereoscopic_video_info_descriptor

descriptor_tag	TS	PS	Identification
55	X	n/a	Transport_profile_descriptor
56	X	n/a	HEVC video descriptor
57 .. 62	n/a	n/a	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Reserved
63	X	X	Extension_descriptor
64 .. 255	n/a	n/a	User Private

2.6.2 Video stream descriptor

The video stream descriptor provides basic information which identifies the coding parameters of a video elementary stream as described in Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 (see Table 2-46).

Table 2-46 – Video stream descriptor

Syntax	No. of bits	Mnemonic
video_stream_descriptor(){		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
multiple_frame_rate_flag	**1**	**bslbf**
frame_rate_code	**4**	**uimsbf**
MPEG_1_only_flag	**1**	**bslbf**
constrained_parameter_flag	**1**	**bslbf**
still_picture_flag	**1**	**bslbf**
if (MPEG_1_only_flag = = '0'){		
profile_and_level_indication	**8**	**uimsbf**
chroma_format	**2**	**uimsbf**
frame_rate_extension_flag	**1**	**bslbf**
reserved	**5**	**bslbf**
}		
}		

2.6.3 Semantic definitions of fields in video stream descriptor

multiple_frame_rate_flag – This 1-bit field when set to '1' indicates that multiple frame rates may be present in the video stream. When set to a value of '0' only a single frame rate is present.

frame_rate_code – This is a 4-bit field as defined in 6.3.3 of Rec. ITU-T H.262 | ISO/IEC 13818-2, except that when the multiple_frame_rate_flag is set to a value of '1' the indication of a particular frame rate also permits certain other frame rates to be present in the video stream, as specified in Table 2-47:

Table 2-47 – Frame rate code

Coded as	Also includes
23.976	
24.0	23.976
25.0	
29.97	23.976
30.0	23.976 24.0 29.97
50.0	25.0
59.94	23.976 29.97
60.0	23.976 24.0 29.97 30.0 59.94

MPEG_1_only_flag – This is a 1-bit field which when set to '1' indicates that the video stream contains only ISO/IEC 11172-2 data. If set to '0' the video stream may contain both Rec. ITU-T H.262 | ISO/IEC 13818-2 video data and constrained parameter ISO/IEC 11172-2 video data.

constrained_parameter_flag – This is a 1-bit field which when set to '1' indicates that the video stream shall not contain unconstrained ISO/IEC 11172-2 video data. If this field is set to '0' the video stream may contain both constrained parameters and unconstrained ISO/IEC 11172-2 video streams. If the MPEG_1_only_flag is set to '0', the constrained_parameter_flag shall be set to '1'.

still_picture_flag – This is a 1-bit field, which when set to '1' indicates that the video stream contains only still pictures. If the bit is set to '0' then the video stream may contain either moving or still picture data.

profile_and_level_indication – This 8-bit field is coded in the same manner as the profile_and_level_indication fields in the Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream. The value of this field indicates a profile and level that is equal to or higher than any profile and level in any sequence in the associated video stream. For the purposes of this subclause, an ISO/IEC 11172-2 constrained parameter stream is considered to be a Main Profile at Low Level stream (MP @ LL).

chroma_format – This 2-bit field is coded in the same manner as the chroma_format fields in the Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream. The value of this field shall be at least equal to or higher than the value of the chroma_format field in any video sequence of the associated video stream. For the purposes of this subclause, an ISO/IEC 11172-2 video stream is considered to have chroma_format field with the value '01', indicating 4:2:0.

frame_rate_extension_flag – This is a 1-bit flag which when set to '1' indicates that either or both the frame_rate_extension_n and the frame_rate_extension_d fields are non-zero in any video sequences of the Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream. For the purposes of this subclause, an ISO/IEC 11172-2 video stream is constrained to have both fields set to zero.

2.6.4 Audio stream descriptor

The audio stream descriptor provides basic information which identifies the coding version of an audio elementary stream as described in ISO/IEC 13818-3 or ISO/IEC 11172-3 (see Table 2-48).

Table 2-48 – Audio stream descriptor

Syntax	No. of bits	Mnemonic
audio_stream_descriptor(){		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
free_format_flag	**1**	**bslbf**
ID	**1**	**bslbf**
layer	**2**	**bslbf**
variable_rate_audio_indicator	**1**	**bslbf**
reserved	**3**	**bslbf**
}		

2.6.5 Semantic definition of fields in audio stream descriptor

free_format_flag – This 1-bit field when set to '1' indicates that the audio stream may contain one or more audio frames with the bitrate_index set to '0000'. If set to '0', then the bitrate_index is not '0000' (refer to 2.4.2.3 of ISO/IEC 13818-3) in any audio frame of the audio stream.

ID – This 1-bit field when set to '1' indicates that the ID field is set to '1' in each audio frame in the audio stream (refer to 2.4.2.3 of ISO/IEC 13818-3).

layer – This 2-bit field is coded in the same manner as the layer field in the ISO/IEC 13818-3 or ISO/IEC 11172-3 audio streams (refer to 2.4.2.3 of ISO/IEC 13818-3). The layer indicated in this field shall be equal to or higher than the highest layer specified in any audio frame of the audio stream.

variable_rate_audio_indicator – This 1-bit flag, when set to '0' indicates that the encoded value of the bit rate field shall not change in consecutive audio frames which are intended to be presented without discontinuity.

2.6.6 Hierarchy descriptor

The hierarchy descriptor provides information to identify the program elements containing components of hierarchically-coded video, audio, and private streams. (See Table 2-49.)

Table 2-49 – Hierarchy descriptor

Syntax	No. of bits	Mnemonic
hierarchy_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
no_view_scalability_flag	**1**	**bslbf**
no_temporal_scalability_flag	**1**	**bslbf**
no_spatial_scalability_flag	**1**	**bslbf**
no_quality_scalability_flag	**1**	**bslbf**
hierarchy_type	**4**	**uimsbf**
reserved	**2**	**bslbf**
hierarchy_layer_index	**6**	**uimsbf**
tref_present_flag	**1**	**bslbf**
reserved	**1**	**bslbf**
hierarchy_embedded_layer_index	**6**	**uimsbf**
reserved	**2**	**bslbf**
hierarchy_channel	**6**	**uimsbf**
}		

2.6.7 Semantic definition of fields in hierarchy descriptor

no_view_scalability_flag – A 1-bit flag, which when set to '0' indicates that the associated program element enhances the number of views of the bit-stream resulting from the program element referenced by the hierarchy_embedded_layer_index. The value of '1' for this flag is reserved.

no_temporal_scalability_flag – A 1-bit flag, which when set to '0' indicates that the associated program element enhances the frame rate of the bit-stream resulting from the program element referenced by the hierarchy_embedded_layer_index. The value of '1' for this flag is reserved.

no_spatial_scalability_flag – A 1-bit flag, which when set to '0' indicates that the associated program element enhances the spatial resolution of the bit-stream resulting from the program element referenced by the hierarchy_embedded_layer_index. The value of '1' for this flag is reserved.

no_quality_scalability_flag – A 1-bit flag, which when set to '0' indicates that the associated program element enhances the SNR quality or fidelity of the bit-stream resulting from the program element referenced by the hierarchy_embedded_layer_index. The value of '1' for this flag is reserved.

hierarchy_type – The hierarchical relation between the associated hierarchy layer and its hierarchy embedded layer is defined in Table 2-50. If scalability applies in more than one dimension, this field shall be set to the value of '8' ("Combined Scalability"), and the flags no_view_scalability_flag, no_temporal_scalability_flag, no_spatial_scalability_flag and no_quality_scalability_flag shall be set accordingly. For MVC video sub-bitstreams, this field shall be set to the value of '9' ("MVC video sub-bitstream") and the flags no_view_scalability_flag, no_temporal_scalability_flag, no_spatial_scalability_flag and no_quality_scalability_flag shall be set to '1'. For MVC base view sub-bitstreams, this field shall be set to the value of '15' and the flags no_view_scalability_flag, no_temporal_scalability_flag, no_spatial_scalability_flag and no_quality_scalability_flag shall be set to '1'. For MVCD video sub-bitstreams, this field shall be set to the value of '9' ("MVCD video sub-bitstream") and the flags no_view_scalability_flag, no_temporal_scalability_flag, no_spatial_scalability_flag and no_quality_scalability_flag shall be set to '1'. For MVCD base view sub-bitstreams, this field shall be set to the value of '15' and the flags no_view_scalability_flag, no_temporal_scalability_flag, no_spatial_scalability_flag and no_quality_scalability_flag shall be set to '1'.

hierarchy_layer_index – The hierarchy_layer_index is a 6-bit field that defines a unique index of the associated program element in a table of coding layer hierarchies. Indices shall be unique within a single program definition. For video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, this is the program element index, which is assigned in a way that the bitstream order will be correct if associated SVC dependency representations of the video sub-bitstreams of the same access unit are re-assembled in increasing order of hierarchy_layer_index. For MVC video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, this is the program element index, which is assigned in a way that the bitstream order will be correct if associated MVC view-component subsets of the MVC video sub-bitstreams of the same access unit are re-assembled in increasing order of hierarchy_layer_index. For MVCD video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, this is the program element index, which is assigned in a way that the bitstream order will be correct if associated MVCD view-component subsets of the MVCD video sub-bitstreams of the same access unit are re-assembled in increasing order of hierarchy_layer_index.

tref_present_flag – A 1-bit flag, which when set to '0' indicates that the TREF field may be present in the PES packet headers in the associated elementary stream. The value of '1' for this flag is reserved.

hierarchy_embedded_layer_index – The hierarchy_embedded_layer_index is a 6-bit field that defines the hierarchy_layer_index of the program element that needs to be accessed and be present in decoding order before decoding of the elementary stream associated with this hierarchy_descriptor. This field is undefined if the hierarchy_type value is 15.

hierarchy_channel – The hierarchy_channel is a 6-bit field that indicates the intended channel number for the associated program element in an ordered set of transmission channels. The most robust transmission channel is defined by the lowest value of this field with respect to the overall transmission hierarchy definition.

NOTE – A given hierarchy_channel may at the same time be assigned to several program elements.

Table 2-50 – Hierarchy_type field values

Value	Description
0	Reserved
1	Spatial Scalability
2	SNR Scalability
3	Temporal Scalability
4	Data partitioning
5	Extension bitstream
6	Private Stream
7	Multi-view Profile
8	Combined Scalability or MV-HEVC sub-partition.
9	MVC video sub-bitstream or MVCD video sub-bitstream
10	Auxiliary picture layer as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2.
11 .. 14	Reserved
15	Base layer or MVC base view sub-bitstream or AVC video sub-bitstream of MVC or HEVC temporal video sub-bitstream or HEVC base sub-partition or Base layer of MVCD base view sub-bitstream or AVC video sub-bitstream of MVCD.

2.6.8 Registration descriptor

The registration_descriptor provides a method to uniquely and unambiguously identify formats of private data (see Table 2-51).

Table 2-51 – Registration descriptor

Syntax	No. of bits	Identifier
registration_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
format_identifier	**32**	**uimsbf**
for (i = 0; i < N; i++){		
additional_identification_info	**8**	**bslbf**
}		
}		

2.6.9 Semantic definition of fields in registration descriptor

format_identifier – The format_identifier is a 32-bit value obtained from a Registration Authority as designated by ISO/IEC JTC 1/SC 29.

additional_identification_info – The meaning of additional_identification_info bytes, if any, are defined by the assignee of that format_identifier, and once defined they shall not change.

2.6.10 Data stream alignment descriptor

The data stream alignment descriptor describes which type of alignment is present in the associated elementary stream. If the data_alignment_indicator in the PES packet header is set to '1' and the descriptor is present, alignment – as specified in this descriptor – is required (see Table 2-52).

Table 2-52 – Data stream alignment descriptor

Syntax	No. of bits	Mnemonic
data_stream_alignment_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
alignment_type	**8**	**uimsbf**
}		

2.6.11 Semantics of fields in data stream alignment descriptor

alignment_type – Table 2-53 describes the alignment type for ISO/IEC 11172-2 video, Rec. ITU-T H.262 | ISO/IEC 13818-2 video, or ISO/IEC 14496-2 visual streams when the data_alignment_indicator in the PES packet header has a value of '1'. For these video streams, the first PES_packet_data_byte following the PES header shall be the first byte of a start code of the type indicated in Table 2-53. At the beginning of a video sequence, the alignment shall occur at the start code of the first sequence header.

NOTE – Specifying alignment type '1' from Table 2-53 does not preclude the alignment from beginning at a GOP or SEQ header.

The definition of an access unit is given in 2.1.1.

Table 2-53 – Video stream alignment values

Alignment type	Description
0	Reserved
1	Slice, or video access unit
2	Video access unit
3	GOP, or SEQ
4	SEQ
5 .. 255	Reserved

Table 2-54 describes the alignment type for Rec. ITU-T H.264 | ISO/IEC 14496-10 video when the data_alignment_indicator in the PES packet header has a value of '1'.

In this case:

- For AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10, the first PES_packet_data_byte following the PES header shall be the first byte of an AVC access unit or the first byte of an AVC slice, as signalled by the alignment_type value.
- For video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, the first PES_packet_data_byte following the PES header shall be the first byte of an SVC dependency representation or the first byte of an SVC slice, as signalled by the alignment_type value.
- For MVC video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, the first PES_packet_data_byte following the PES header shall be the first byte of an MVC view-component subset or the first byte of an MVC slice, as signalled by the alignment_type value.
- For MVCD video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, the first PES_packet_data_byte following the PES header shall be the first byte of an MVCD view-component subset, the first byte of an MVC slice or the first byte of MVCD slice, as signalled by the alignment_type value.

Table 2-54 – AVC video stream alignment values

Alignment type	Description
0	Reserved
1	AVC slice or AVC access unit
2	AVC access unit
3	SVC slice or SVC dependency representation
4	SVC dependency representation
5	MVC slice or MVC view-component subset
6	MVC view-component subset
7	MVCD slice or MVCD view-component subset
8	MVCD view-component subset
9 .. 255	Reserved

Table 2-55 describes the alignment type for HEVC when the data_alignment_indicator in the PES packet header has a value of '1'.

Table 2-55 – HEVC video stream alignment values

Alignment type	Description
0	Reserved
1	HEVC access unit
2	HEVC slice
3	HEVC access unit or slice
4	HEVC tile of slices
5	HEVC access unit or tile of slices
6	HEVC slice or tile of slices
7	HEVC access unit or slice or tile of slices
8	HEVC slice segment
9	HEVC slice segment or access unit
10	HEVC slice segment or slice
11	HEVC slice segment or access unit or slice
12	HEVC slice segment or tile of slices
13	HEVC slice segment or access unit or tile of slices
14	HEVC slice segment or slice or tile of slices
15	HEVC slice segment or access unit or slice or tile of slices
16 .. 255	Reserved

Table 2-56 describes the audio alignment type when the data_alignment_indicator in the PES packet header has a value of '1'. In this case the first PES_packet_data_byte following the PES header is the first byte of an audio sync word.

Table 2-56 – Audio stream alignment values

Alignment type	Description
0	Reserved
1	Sync word
2 .. 255	Reserved

2.6.12 Target background grid descriptor

It is possible to have one or more video streams which, when decoded, are not intended to occupy the full display area (e.g., a monitor). The combination of target_background_grid_descriptor and video_window_descriptors allows the display of these video windows in their desired locations. The target_background_grid_descriptor is used to describe a grid of unit pixels projected on to the display area. The video_window_descriptor is then used to describe, for the associated stream, the location on the grid at which the top left pixel of the display window or display rectangle of the video presentation unit should be displayed. This is represented in Figure 2-3.

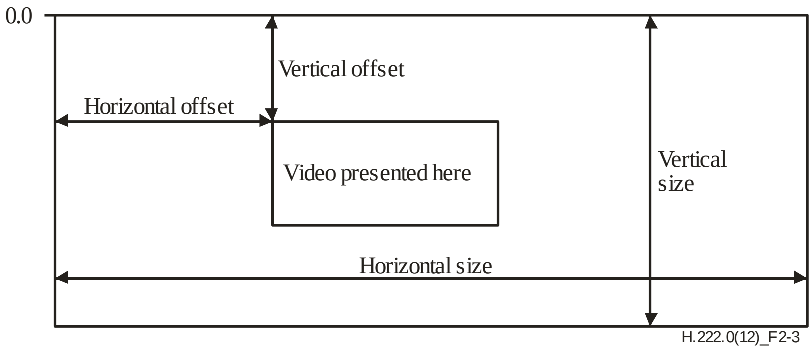


Figure 2-3 – Target background grid descriptor display area

2.6.13 Semantics of fields in target background grid descriptor

horizontal_size – The horizontal size of the target background grid in pixels.

vertical_size – The vertical size of the target background grid in pixels.

aspect_ratio_information – Specifies the sample aspect ratio or display aspect ratio of the target background grid. Aspect_ratio_information is defined in Rec. ITU-T H.262 | ISO/IEC 13818-2 (see Table 2-57).

Table 2-57 – Target background grid descriptor

Syntax	No. of bits	Mnemonic
target_background_grid_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
horizontal_size	**14**	**uimsbf**
vertical_size	**14**	**uimsbf**
aspect_ratio_information	**4**	**uimsbf**
}		

2.6.14 Video window descriptor

The video window descriptor (see Table 2-58) is used to describe the window characteristics of the associated video elementary stream. Its values reference the target background grid descriptor for the same stream. Also see target_background_grid_descriptor in 2.6.12.

Table 2-58 – Video window descriptor

Syntax	No. of bits	Mnemonic
video_window_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
horizontal_offset	**14**	**uimsbf**
vertical_offset	**14**	**uimsbf**
window_priority	**4**	**uimsbf**
}		

2.6.15 Semantic definition of fields in video window descriptor

horizontal_offset – The value indicates the horizontal position of the top left pixel of the current video display window or display rectangle if indicated in the picture display extension on the target background grid for display as defined in the target_background_grid_descriptor. The top left pixel of the video window shall be one of the pixels of the target background grid (refer to Figure 2-3).

vertical_offset – The value indicates the vertical position of the top left pixel of the current video display window or display rectangle if indicated in the picture display extension on the target background grid for display as defined in the target_background_grid_descriptor. The top left pixel of the video window shall be one of the pixels of the target background grid (refer to Figure 2-3).

window_priority – The value indicates how windows overlap. A value of 0 being lowest priority and a value of 15 is the highest priority, i.e., windows with priority 15 are always visible.

2.6.16 Conditional access descriptor

The conditional access descriptor is used to specify both system-wide conditional access management information such as EMMs and elementary stream-specific information such as ECMs. It may be used in both the TS_program_map_section (refer to 2.4.4.8) and the program_stream_map (refer to 2.5.3). If any elementary stream is scrambled, a CA descriptor shall be present for the program containing that elementary stream. If any system-wide conditional access management information exists within a transport stream, a CA descriptor shall be present in the conditional access table.

When the CA descriptor is found in the TS_program_map_section (table_id = 0x02), the CA_PID points to packets containing program related access control information, such as ECMs. Its presence as program information indicates applicability to the entire program. In the same case, its presence as extended ES information indicates applicability to the associated program element. Provision is also made for private data.

When the CA descriptor is found in the CA_section (table_id = 0x01), the CA_PID points to packets containing system-wide and/or access control management information, such as EMMs.

The contents of the transport stream packets containing conditional access information are privately defined (see Table 2-59).

Table 2-59 – Conditional access descriptor

Syntax	No. of bits	Mnemonic
CA_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
CA_system_ID	**16**	**uimsbf**
reserved	**3**	**bslbf**
CA_PID	**13**	**uimsbf**
for (i = 0; i < N; i++) {		
private_data_byte	**8**	**uimsbf**
}		
}		

2.6.17 Semantic definition of fields in conditional access descriptor

CA_system_ID – This is a 16-bit field indicating the type of CA system applicable for either the associated ECM and/or EMM streams. The coding of this is privately defined and is not specified by ITU-T | ISO/IEC.

CA_PID – This is a 13-bit field indicating the PID of the transport stream packets which shall contain either ECM or EMM information for the CA systems as specified with the associated CA_system_ID. The contents (ECM or EMM) of the packets indicated by the CA_PID is determined from the context in which the CA_PID is found, i.e., a TS_program_map_section or the CA table in the transport stream, or the stream_id field in the program stream.

In transport streams, the presence of PID 0x03 indicates that there is IPMP as described in ISO/IEC 13818-11 used by components in the transport stream. In program streams, the presence of stream_ID_extension value 0x00 indicates that IPMP as described in ISO/IEC 13818-11 is used by components in the program stream. Within a given Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, components could use both IPMP as described in ISO/IEC 13818-11 as well as CA as defined in ISO/IEC 13818-1:2006. Compatibility between the two schemes is described in ISO/IEC 13818-11.

2.6.18 ISO 639 language descriptor

The language descriptor is used to specify the language of the associated program element (see Table 2-60).

Table 2-60 – ISO 639 language descriptor

Syntax	No. of bits	Mnemonic
ISO_639_language_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
for (i = 0; i < N; i++) {		
ISO_639_language_code	**24**	**bslbf**
audio_type	**8**	**bslbf**
}		
}		

2.6.19 Semantic definition of fields in ISO 639 language descriptor

ISO_639_language_code – Identifies the language or languages used by the associated program element. The ISO_639_language_code contains a 3-character code as specified by ISO 639, Part 2. Each character is coded into 8 bits according to ISO 8859-1 and inserted in order into this 24-bit field. In the case of multilingual audio streams the sequence of ISO_639_language_code fields shall reflect the content of the audio stream.

audio_type – The audio_type is an 8-bit field which specifies the type of stream defined in Table 2-61.

Table 2-61 – Audio type values

Value	Description
0x00	Undefined
0x01	Clean effects[a]
0x02	Hearing impaired[b]
0x03	Visual impaired commentary[c]
0x04 .. 0x7F	User Private
0x80 .. 0xFF	Reserved

[a] Clean effects: This value indicates that the referenced program element has no language.
[b] Hearing impaired: This value indicates that the referenced program element is prepared for the hearing impaired.
[c] Visual_impaired_commentary: This value indicates that the referenced program element is prepared for the visually impaired viewer.

2.6.20 System clock descriptor

This descriptor conveys information about the system clock that was used to generate the timestamps.

If an external clock reference was used, the external_clock_reference_indicator may be set to '1'. The decoder optionally may use the same external reference if it is available.

If the system clock is more accurate than the 30-ppm accuracy required, then the accuracy of the clock can be communicated by encoding it in the clock_accuracy fields. The clock frequency accuracy is:

$$clock\_accuracy\_integer \times 10^{-clock\_accuracy\_exponent}\ ppm \tag{2-26}$$

If clock_accuracy_integer is set to '0', then the system clock accuracy is 30 ppm. When the external_clock_reference_indicator is set to '1', the clock accuracy pertains to the external reference clock (see Table 2-62).

Table 2-62 – System clock descriptor

Syntax	No. of bits	Mnemonic
system_clock_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
external_clock_reference_indicator	**1**	**bslbf**
reserved	**1**	**bslbf**
clock_accuracy_integer	**6**	**uimsbf**
clock_accuracy_exponent	**3**	**uimsbf**
reserved	**5**	**bslbf**
}		

2.6.21 Semantic definition of fields in system clock descriptor

external_clock_reference_indicator – This is a 1-bit indicator. When set to '1', it indicates that the system clock has been derived from an external frequency reference that may be available at the decoder.

clock_accuracy_integer – This is a 6-bit integer. Together with the clock_accuracy_exponent, it gives the fractional frequency accuracy of the system clock in parts per million.

clock_accuracy_exponent – This is a 3-bit integer. Together with the clock_accuracy_integer, it gives the fractional frequency accuracy of the system clock in parts per million.

2.6.22 Multiplex buffer utilization descriptor

The multiplex buffer utilization descriptor provides bounds on the occupancy of the STD multiplex buffer. This information is intended for devices such as remultiplexers, which may use this information to support a desired re-multiplexing strategy (see Table 2-63).

Table 2-63 – Multiplex buffer utilization descriptor

Syntax	No. of bits	Mnemonic
Multiplex_buffer_utilization_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
bound_valid_flag	**1**	**bslbf**
LTW_offset_lower_bound	**15**	**uimsbf**
reserved	**1**	**bslbf**
LTW_offset_upper_bound	**15**	**uimsbf**
}		

2.6.23 Semantic definition of fields in multiplex buffer utilization descriptor

bound_valid_flag – A value of '1' indicates that the LTW_offset_lower_bound and the LTW_offset_upper_bound fields are valid.

LTW_offset_lower_bound – This 15-bit field is defined only if the bound_valid flag has a value of '1'. When defined, this field has the units of (27 MHz/300) clock periods, as defined for the LTW_offset (refer to 2.4.3.4). The LTW_offset_lower_bound represents the lowest value that any LTW_offset field would have, if that field were coded in every packet of the stream or streams referenced by this descriptor. Actual LTW_offset fields may or may not be coded in the bitstream when the multiplex buffer utilization descriptor is present. This bound is valid until the next occurrence of this descriptor.

LTW_offset_upper_bound – This 15-bit field is defined only if the bound_valid has a value of '1'. When defined, this field has the units of (27 MHz/300) clock periods, as defined for the LTW_offset (refer to 2.4.3.4). The LTW_offset_upper_bound represents the largest value that any LTW_offset field would have, if that field were coded in every packet of the stream or streams referenced by this descriptor. Actual LTW_offset fields may or may not be coded in the bitstream when the multiplex buffer utilization descriptor is present. This bound is valid until the next occurrence of this descriptor.

2.6.24 Copyright descriptor

The copyright_descriptor provides a method to enable audiovisual works identification. This copyright_descriptor applies to programs or program elements within programs (see Table 2-64).

Table 2-64 – Copyright descriptor

Syntax	No. of bits	Identifier
copyright_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
copyright_identifier	**32**	**uimsbf**
for (i = 0; i < N; i++){		
additional_copyright_info	**8**	**bslbf**
}		
}		

2.6.25 Semantic definition of fields in copyright descriptor

copyright_identifier – This field is a 32-bit value obtained from the Registration Authority.

additional_copyright_info – The meaning of additional_copyright_info bytes, if any, are defined by the assignee of that copyright_identifier, and once defined, they shall not change.

2.6.26 Maximum bitrate descriptor

See Table 2-65.

Table 2-65 – Maximum bitrate descriptor

Syntax	No. of bits	Identifier
maximum_bitrate_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
reserved	**2**	**bslbf**
maximum_bitrate	**22**	**uimsbf**
}		

2.6.27 Semantic definition of fields in maximum bitrate descriptor

maximum_bitrate – The maximum bitrate is coded as a 22-bit positive integer in this field. The value indicates an upper bound of the bitrate, including transport overhead, that will be encountered in this program element or program. The value of maximum_bitrate is expressed in units of 50 bytes/second. The maximum_bitrate_descriptor is included in the Program Map Table (PMT). Its presence as extended program information indicates applicability to the entire program. Its presence as ES information indicates applicability to the associated program element.

2.6.28 Private data indicator descriptor

See Table 2-66.

Table 2-66 – Private data indicator descriptor

Syntax	No. of bits	Identifier
private_data_indicator_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
private_data_indicator	**32**	**uimsbf**
}		

2.6.29 Semantic definition of fields in Private data indicator descriptor

private_data_indicator – The value of the private_data_indicator is private and shall not be defined by ITU-T | ISO/IEC.

2.6.30 Smoothing buffer descriptor

This descriptor is optional and conveys information about the size of a smoothing buffer, SB_n, associated with this descriptor, and the associated leak rate out of that buffer, for the program element(s) that it refers to.

In the case of transport streams, bytes of transport stream packets of the associated program element(s) present in the transport stream are input to a buffer SB_n of size given by sb_size, at the time defined by equation 2-4.

In the case of program streams, bytes of all PES packets of the associated elementary streams, are input to a buffer SB_n of size given by sb_size, at the time defined by equation 2-21.

When there is data present in this buffer, bytes are removed from this buffer at a rate defined by sb_leak_rate. The buffer, SB_n shall never overflow. During the continuous existence of a program, the value of the elements of the Smoothing Buffer descriptor of the different program element(s) in the program, shall not change.

The meaning of the smoothing buffer_descriptor is only defined when it is included in the PMT or the program stream Map.

If, in the case of a transport stream, it is present in the ES info in the Program Map Table, all transport stream packets of the PID of that program element enter the smoothing buffer.

If, in the case of a transport stream, it is present in the program information, the following transport stream packets enter the smoothing buffer:

- all transport stream packets of all PIDs listed as elementary_PIDs in the extended program information as well as;
- all transport stream packets of the PID which is equal to the PMT_PID of this section;
- all transport stream packets of the PCR_PID of the program.

All bytes that enter the associated buffer also exit it.

At any given time there shall be at most one descriptor referring to any individual program element and at most one descriptor referring to the program in its entirety.

Table 2-67 – Smoothing buffer descriptor

Syntax	No. of bits	Mnemonic
smoothing_buffer_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
reserved	**2**	**bslbf**
sb_leak_rate	**22**	**uimsbf**
reserved	**2**	**bslbf**
sb_size	**22**	**uimsbf**
}		

2.6.31 Semantic definition of fields in smoothing buffer descriptor

sb_leak_rate – This 22-bit field is coded as a positive integer. Its contents indicate the value of the leak rate out of the SB_n buffer for the associated elementary stream or other data in units of 400 bits/s.

sb_size – This 22-bit field is coded as a positive integer. Its contents indicate the value of the size of the multiplexing buffer smoothing buffer SB_n for the associated elementary stream or other data in units of 1 byte (see Table 2-67).

2.6.32 STD descriptor

This descriptor is optional and applies only to the T-STD model and to Rec. ITU-T H.262 | ISO/IEC 13818-2 video elementary streams, and is used as specified in 2.4.2. This descriptor does not apply to program streams (see Table 2-68).

Table 2-68 – STD descriptor

Syntax	No. of bits	Mnemonic
STD_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
reserved	**7**	**bslbf**
leak_valid_flag	**1**	**bslbf**
}		

2.6.33 Semantic definition of fields in STD descriptor

leak_valid_flag – The leak_valid_flag is a 1-bit flag. When set to '1', the transfer of data from the buffer MB_n to the buffer EB_n in the T-STD uses the leak method as defined in 2.4.2.4. If this flag has a value equal to '0', and the vbv_delay fields present in the associated video stream do not have the value 0xFFFF, the transfer of data from the buffer MB_n to the buffer EB_n uses the vbv_delay method as defined in 2.4.2.4.

2.6.34 IBP descriptor

This optional descriptor provides information about some characteristics of the sequence of frame types in an ISO/IEC 11172-2, Rec. ITU-T H.262 | ISO/IEC 13818-2, or ISO/IEC 14496-2 video stream (see Table 2-69).

Table 2-69 – IBP descriptor

Syntax	No. of bits	Mnemonic
ibp_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
closed_gop_flag	**1**	**uimsbf**
identical_gop_flag	**1**	**uimsbf**
max_gop-length	**14**	**uimsbf**
}		

2.6.35 Semantic definition of fields in IBP descriptor

closed_gop_flag – This 1-bit flag when set to '1' indicates that a group of pictures header is encoded before every I-frame and that the closed_gop flag is set to '1' in all group of pictures headers in the video sequence.

identical_gop_flag – This 1-bit flag when set to '1' indicates that the number of P-frames and B-frames between I-frames, and the picture coding types and sequence of picture types between I-pictures is the same throughout the sequence, except possibly for the pictures up to the second I-picture.

max_gop_length – This 14-bit unsigned integer indicates the maximum number of the coded pictures between any two consecutive I-pictures in the sequence. The value of '0' is forbidden.

2.6.36 MPEG-4 video descriptor

For individual ISO/IEC 14496-2 streams directly carried in PES packets, as defined in 2.11.2, the MPEG-4 video descriptor (see Table 2-70) provides basic information for identifying the coding parameters of such visual elementary streams. The MPEG-4 video descriptor does not apply to ISO/IEC 14496-2 streams encapsulated in SL-packets and in FlexMux packets, as defined in 2.11.3.

Table 2-70 – MPEG-4 video descriptor

Syntax	No. of bits	Mnemonic
MPEG-4_video_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
MPEG-4_visual_profile_and_level	**8**	**uimsbf**
}		

2.6.37 Semantic definition of fields in MPEG-4 video descriptor

MPEG-4_video_profile_and_level – This 8-bit field shall identify the profile and level of the ISO/IEC 14496-2 video stream. This field shall be coded with the same value as the profile_and_level_indication field in the Visual Object Sequence Header in the associated ISO/IEC 14496-2 stream.

2.6.38 MPEG-4 audio descriptor

For individual ISO/IEC 14496-3 streams directly carried in PES packets, as defined in 2.11.2, the MPEG-4 audio descriptor (see Table 2-71) provides basic information for identifying the coding parameters of such audio elementary streams. The MPEG-4 audio descriptor does not apply to ISO/IEC 14496-3 streams encapsulated in SL-packets and in FlexMux packets, as defined in 2.11.3.

Table 2-71 – MPEG-4 audio descriptor

Syntax	No. of bits	Mnemonic
MPEG-4_audio_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
MPEG-4_audio_profile_and_level	**8**	**uimsbf**
}		

2.6.39 Semantic definition of fields in MPEG-4 audio descriptor

MPEG-4_audio_profile_and_level – This 8-bit field identifies the profile and level of the ISO/IEC 14496-3 audio stream corresponding to Table 2-72. If encoded with the value 0x0F, then it is signalled that no profile and level is defined for the associated MPEG-4 audio stream. The encoded value 0xFF indicates that the audio profile and level is not specified by the MPEG-4_audio_profile_and_level field; in that case, in addition to the MPEG-4 audio descriptor, an MPEG-4 audio extension descriptor shall be associated with the same MPEG-4 audio stream. In all other cases, next to an MPEG-4 audio descriptor, also an MPEG-4 audio extension descriptor may be associated with the same MPEG-4 audio stream.

Table 2-72 – MPEG-4_audio_profile_and_level assignment values

Value	Description
0x00 .. 0x0E	Reserved
0x0F	No audio profile and level defined for the associated MPEG-4 audio stream
0x10	Main profile, level 1
0x11	Main profile, level 2
0x12	Main profile, level 3
0x13	Main profile, level 4
0x14 .. 0x17	Reserved
0x18	Scalable Profile, level 1
0x19	Scalable Profile, level 2
0x1A	Scalable Profile, level 3
0x1B	Scalable Profile, level 4
0x1C .. 0x1F	Reserved
0x20	Speech profile, level 1
0x21	Speech profile, level 2
0x22 .. 0x27	Reserved
0x28	Synthesis profile, level 1
0x29	Synthesis profile, level 2
0x2A	Synthesis profile, level 3
0x2B .. 0x2F	Reserved

Value	Description
0x30	High quality audio profile, level 1
0x31	High quality audio profile, level 2
0x32	High quality audio profile, level 3
0x33	High quality audio profile, level 4
0x34	High quality audio profile, level 5
0x35	High quality audio profile, level 6
0x36	High quality audio profile, level 7
0x37	High quality audio profile, level 8
0x38	Low delay audio profile, level 1
0x39	Low delay audio profile, level 2
0x3A	Low delay audio profile, level 3
0x3B	Low delay audio profile, level 4
0x3C	Low delay audio profile, level 5
0x3D	Low delay audio profile, level 6
0x3E	Low delay audio profile, level 7
0x3F	Low delay audio profile, level 8
0x40	Natural audio profile, level 1
0x41	Natural audio profile, level 2
0x42	Natural audio profile, level 3
0x43	Natural audio profile, level 4
0x44 .. 0x47	Reserved
0x48	Mobile audio internetworking profile, level 1
0x49	Mobile audio internetworking profile, level 2
0x4A	Mobile audio internetworking profile, level 3
0x4B	Mobile audio internetworking profile, level 4
0x4C	Mobile audio internetworking profile, level 5
0x4D	Mobile audio internetworking profile, level 6
0x4E .. 0x4F	Reserved
0x50	AAC profile, level 1
0x51	AAC profile, level 2
0x52	AAC profile, level 4
0x53	AAC profile, level 5
0x54	AAC profile, level 6
0x55	AAC profile, level 7
0x56 .. 0x57	Reserved
0x58	High efficiency AAC profile, level 2
0x59	High efficiency AAC profile, level 3
0x5A	High efficiency AAC profile, level 4
0x5B	High efficiency AAC profile, level 5
0x5C	High efficiency AAC profile, level 6
0x5D	High efficiency AAC profile, level 7
0x5E .. 0x5F	Reserved
0x60	High efficiency AAC v2 profile, level 2
0x61	High efficiency AAC v2 profile, level 3
0x62	High efficiency AAC v2 profile, level 4
0x63	High efficiency AAC v2 profile, level 5
0x64	High efficiency AAC v2 profile, level 6
0x65	High efficiency AAC v2 profile, level 7
0x66 .. 0x67	Reserved
0x68	Extended HE AAC Profile, level 1
0x69	Extended HE AAC Profile, level 2
0x6A	Extended HE AAC Profile, level 3

Value	Description
0x6B	Extended HE AAC Profile, level 4
0x6C	Extended HE AAC Profile, level 6
0x6D	Extended HE AAC Profile, level 7
0x6E .. 0x6F	Reserved
0x70	Baseline USAC Profile, level 1
0x71	Baseline USAC Profile, level 2
0x72	Baseline USAC Profile, level 3
0x73	Baseline USAC Profile, level 4
0x74 .. 0x7F	Reserved
0x80	Low Delay AAC Profile, level 1
0x81 .. 0x87	Reserved
0x88	Low Delay AAC v2 Profile, level 1
0x89	Low Delay AAC v2 Profile, level 2
0x8A	Low Delay AAC v2 Profile, level 3
0x8B	Low Delay AAC v2 Profile, level 4
0x8C .. 0x8F	Reserved
0x90	High Definition AAC Profile, level 1
0x91 .. 0x97	Reserved
0x98	ALS Simple Profile, level 1
0x99 .. 0xFE	Reserved
0xFF	Audio profile and level not specified by the MPEG-4_audio_profile_and_level field in this descriptor

2.6.40 IOD descriptor

The IOD descriptor (see Table 2-73) encapsulates the InitialObjectDescriptor structure. An initial object descriptor allows access to a set of ISO/IEC 14496 streams by identifying the ES_ID values of the ISO/IEC 14496-1 scene description and object descriptor streams. Both the scene description stream and the object descriptor stream contain further information about the ISO/IEC 14496 streams that are part of the scene. See Annex R for a description of the content access procedure. The InitialObjectDescriptor is specified in 8.6.3 of ISO/IEC 14496-1.

Within a transport stream, the IOD descriptor shall be conveyed in the descriptor loop immediately following the program_info_length field in the Program Map Table. If a program stream map is present in a program stream, the IOD descriptor shall be conveyed in the descriptor loop immediately following the program_stream_info_length field in the program stream map. More than one IOD descriptor may be associated with a program.

NOTE – This Specification does not specify how the IOD_label may be used by higher level service information to uniquely select one of the ISO/IEC 14496 presentations identified by multiple IOD descriptors.

Table 2-73 – IOD descriptor

Syntax	No. of bits	Mnemonic
IOD_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
Scope_of_IOD_label	**8**	**uimsbf**
IOD_label	**8**	**uimsbf**
InitialObjectDescriptor ()		
}		

2.6.41 Semantic definition of fields in IOD descriptor

Scope_of_IOD_label – This 8-bit field specifies the scope of the IOD_label field. A value of 0x10 indicates that the IOD_label is unique within the program stream or within the specific program in a transport stream in which the IOD descriptor is carried. A value of 0x11 indicates that the IOD_label is unique within the transport stream in which the IOD descriptor is carried. All other values of the Scope_of_IOD_label field are reserved.

IOD_label – This 8-bit field specifies the label of the IOD descriptor.

InitialObjectDescriptor () – This structure is defined in 8.6.3.1 of ISO/IEC 14496-1.

2.6.42 SL descriptor

The SL descriptor (see Table 2-74) shall be used when a single ISO/IEC 14496-1 SL-packetized stream is encapsulated in PES packets. The SL descriptor associates the ES_ID of this SL-packetized stream to an elementary_PID in case of a transport stream or to an elementary_stream_id in case of a program stream. Within a transport stream, the SL descriptor shall be conveyed for the corresponding elementary stream in the descriptor loop immediately following the ES_info_length field in the Program Map Table. If a program stream map is present in a program stream, the SL descriptor shall be conveyed in the descriptor loop immediately following the elementary_stream_info_length field within the Program Stream Map.

NOTE – SL packetized streams may be used in a program stream. However, only one stream_id exists for ISO/IEC 14496-1 SL-packetized streams. In order to associate multiple such streams within a program stream to an ISO/IEC 14496-1 scene, FlexMux has to be used and signalled appropriately by an FMC descriptor. This limitation does not exist in a transport stream where the SL descriptor provides unambiguous mapping between an ISO/IEC 14496-1 ES_ID value and a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 elementary_PID value.

Table 2-74 – SL descriptor

Syntax	No. of bits	Mnemonic
SL_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
ES_ID	**16**	**uimsbf**
}		

2.6.43 Semantic definition of fields in SL descriptor

ES_ID – This 16-bit field shall specify the identifier of an ISO/IEC 14496-1 SL-packetized stream.

2.6.44 FMC descriptor

The FMC descriptor (see Table 2-75) indicates that the ISO/IEC 14496-1 FlexMux tool has been used to multiplex ISO/IEC 14496-1 SL-packetized streams into a FlexMux stream before encapsulation in PES packets or ISO//IEC14496_sections. The FMC descriptor associates FlexMux channels to the ES_ID values of the SL-packetized streams in the FlexMux stream.

An FMC descriptor is required for each program element referenced by an elementary_PID value in a transport stream and for each elementary_stream_id in a program stream that conveys a FlexMux stream. Within a transport stream, the FMC descriptor shall be conveyed for the corresponding elementary stream in the descriptor loop immediately following the ES_info_length field in the Program Map Table. If a Program Stream Map is present in a program stream, the FMC descriptor shall be conveyed in the descriptor loop immediately following the elementary_stream_info_length field in the program stream Map.

For each SL_packetized stream in a FlexMux stream, the FlexMux channel shall be identified by a single entry in the FMC descriptor.

Table 2-75 – FMC descriptor

Syntax	No. of bits	Mnemonic
FMC_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
for (i = 0; i < descriptor_length; i + = 3) {		
ES_ID	**16**	**uimsbf**
FlexMuxChannel	**8**	**uimsbf**
}		
}		

2.6.45 Semantic definition of fields in FMC descriptor

ES_ID – This 16-bit field specifies the identifier of an ISO/IEC 14496-1 SL-packetized stream.

FlexMuxChannel – This 8-bit field specifies the number of the FlexMux channel used for this SL-packetized stream.

2.6.46 External_ES_ID descriptor

The External_ES_ID descriptor (see Table 2-76) assigns an ES_ID, as defined in ISO/IEC 14496-1, to a program element to which no ES_ID value has been assigned by other means. This ES_ID allows reference to a non-ISO/IEC 14496 component in the scene description or, for example, to associate a non-ISO/IEC 14496 component with an IPMP stream.

Within a transport stream, the assignment of an ES_ID shall be made by conveying an External_ES_ID descriptor for the corresponding elementary stream in the descriptor loop immediately following the ES_info_length field in the Program Map Table. If a program stream map is present in a program stream, the External_ES_ID descriptor shall be conveyed in the descriptor loop immediately following the elementary_stream_info_length field in the Program Stream Map.

Table 2-76 – External_ES_ID descriptor

Syntax	No. of bits	Mnemonic
External_ES_ID_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
External_ES_ID	**16**	**uimsbf**
}		

2.6.47 Semantic definition of fields in External_ES_ID descriptor

External_ES_ID – This 16-bit field assigns an ES_ID identifier, as defined in ISO/IEC 14496-1, to a component of a program.

2.6.48 Muxcode descriptor

The Muxcode descriptor (see Table 2-77) conveys MuxCodeTableEntry structures as defined in 11.2.4.3 of ISO/IEC 14496-1. MuxCodeTableEntries configure the MuxCode mode of FlexMux.

One or more Muxcode descriptors may be associated with each elementary_PID or elementary_stream_id, respectively, conveying an ISO/IEC 14496-1 FlexMux stream that utilizes the MuxCode mode. Within a transport stream, the Muxcode descriptor shall be conveyed for the corresponding elementary stream in the descriptor loop immediately following the ES_info_length field in the Program Map Table. If a Program Stream Map is present in a program stream, the Muxcode descriptor shall be conveyed in the descriptor loop immediately following the elementary_stream_info_length field in the Program Stream Map.

MuxCodeTableEntries may be updated with new versions. In case of such updates, the version_number of each Program Map Table or the program_stream_map_version of each Program Stream Map, respectively, carrying the MuxCode descriptor in their descriptor loop shall be incremented by 1 modulo 32.

Table 2-77 – Muxcode descriptor

Syntax	No. of bits	Mnemonic
Muxcode_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
for (i = 0; i < N; i++) {		
MuxCodeTableEntry ()		
}		
}		

2.6.49 Semantic definition of fields in Muxcode descriptor

MuxCodeTableEntry () – This structure is defined in 11.2.4.3 of ISO/IEC 14496-1.

2.6.50 FmxBufferSize descriptor

The FmxBufferSize descriptor (see Table 2-78) conveys the size of the FlexMux buffer (FB) for each SL packetized stream multiplexed in a FlexMux stream.

One FmxBufferSize descriptor shall be associated with each elementary_PID or elementary_stream_id, respectively, conveying an ISO/IEC 14496-1 FlexMux stream. Within a transport stream, the FmxBufferSize descriptor shall be conveyed for the corresponding elementary stream in the descriptor loop immediately following the ES_info_length field in the Program Map Table. If a Program Stream Map is present in a program stream, the FmxBufferSize descriptor shall be conveyed in the descriptor loop immediately following the elementary_stream_info_length field within the Program Stream Map.

Table 2-78 – FmxBufferSize descriptor

Syntax	No. of bits	Mnemonic
FmxBufferSize_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
DefaultFlexMuxBufferDescriptor()		
for (i=0; i<descriptor_length; i += 4) {		
FlexMuxBufferDescriptor()		
}		
}		

2.6.51 Semantic definition of fields in FmxBufferSize descriptor

DefaultFlexMuxBufferDescriptor() – This descriptor specifies the default FlexMux buffer size for this FlexMux stream. It is defined in 11.2 of ISO/IEC 14496-1.

FlexMuxBufferDescriptor() – This descriptor specifies the FlexMux buffer size for one SL-packetized stream carried within the FlexMux stream. It is defined in 11.2 of ISO/IEC 14496-1.

2.6.52 MultiplexBuffer descriptor

The MultiplexBuffer descriptor (see Table 2-79) conveys the size of the multiplex buffer MB_n, as well as the leak rate Rx_n at which data is transferred from transport buffer TB_n into buffer MB_n for a specific Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program element referenced by an elementary_PID value in the Program Map Table.

One MultiplexBuffer descriptor shall be associated with each elementary_PID that contains an ISO/IEC 14496 FlexMux stream or SL-packetized stream, including those containing ISO_IEC_14496_sections. See 2.11.3.9 for the definition of buffers and rates in the T-STD model for decoding of ISO/IEC 14496 content.

The MultiplexBuffer descriptor shall be conveyed in the descriptor loop immediately following the ES_info_length field in the Program Map Table.

Table 2-79 – MultiplexBuffer descriptor

Syntax	No. of bits	Mnemonic
MultiplexBuffer_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
MB_buffer_size	**24**	**uimsbf**
TB_leak_rate	**24**	**uimsbf**
}		

2.6.53 Semantic definition of fields in MultiplexBuffer descriptor

MB_buffer_size – This 24-bit field shall specify the size in byte of buffer MB_n of the elementary stream n that is associated with this descriptor.

TB_leak_rate – This 24-bit field shall specify in units of 400 bits per second the rate at which data is transferred from transport buffer TB_n to multiplex buffer MB_n for the elementary stream n that is associated with this descriptor.

2.6.54 FlexMuxTiming descriptor

See Table 2-80.

Table 2-80 – FlexMuxTiming descriptor

Syntax	No. of bits	Mnemonic
FlexMuxTiming_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
FCR_ES_ID	**16**	**uimsbf**
FCRResolution	**32**	**uimsbf**
FCRLength	**8**	**uimsbf**
FmxRateLength	**8**	**uimsbf**
}		

2.6.55 Semantic definition of fields in FlexMuxTiming descriptor

FCR_ES_ID – is the ES_ID associated with this clock reference stream

FCRResolution – is the resolution of the object time base in cycles per second

FCRLength – is the length of the fmxClockReference field in FlexMux packets with index = 238 A length of zero shall indicate that no FlexMux packets with index = 238 are present in this FlexMux stream. FCRlength shall take values between zero and 64.

FmxRateLength – Is the length of the fmxRate field in FlexMux packets with index = 238? FmxRateLength shall take values between 1 and 32.

2.6.56 Content labelling descriptor

The content labelling descriptor assigns a label to content; the label can be used by metadata to reference the associated content. This label, the content_reference_id_record, is metadata application format specific. The content labelling descriptor is associated with a content segment. For the purpose of this clause, a content segment is defined as a portion in time of a program, an elementary stream (such as audio or video) or any combination of programs or elementary streams. The descriptor may be included in the PMT in the descriptor loop for either the program or an elementary stream, but may also be contained in tables not defined in this Specification, for example tables to describe segments of programs or elementary streams. The content labelling descriptor also provides information on which content time base is used and on the offset between the content time base and the metadata time base. When the Normal Play Time (NPT) concept of DSM-CC, as specified in ISO/IEC 13818-6, is used as the content time base, the ID of the NPT time base is provided. The descriptor allows for carriage of private data. See Table 2-81.

Table 2-81 – Content labelling descriptor

Syntax	No. of bits	Mnemonic
Content_labeling_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
metadata_application_format	**16**	**uimsbf**
if (metadata_application_format== 0xFFFF){		
metadata_application_format_identifier	**32**	**uimsbf**
}		
content_reference_id_record_flag	**1**	**bslbf**
content_time_base_indicator	**4**	**uimsbf**
reserved	**3**	**bslbf**
if (content_reference_id_record_flag == '1'){		
content_reference_id_record_length	**8**	**uimsbf**
for (i=0; i<content_reference_id_record_length;i++){		
content_reference_id_byte	**8**	**bslbf**
}		
}		
if (content_time_base_indicator == 1 || content_time_base_indicator == 2){		
reserved	**7**	**bslbf**
content_time_base_value	**33**	**uimsbf**
reserved	**7**	**bslbf**
metadata_time_base_value	**33**	**uimsbf**
}		
if (content_time_base_indicator== 2){		
reserved	**1**	**bslbf**
contentId	**7**	**uimsbf**
}		
if (content_time_base_indicator ≥ 3 && content_time_base_indicator ≤ 7){		
time_base_association_data_length	**8**	**uimsbf**
for (i=0; i< time_base_association_data_length;i++){		
reserved	**8**	**bslbf**
}		
}		
for (i=0; i<N;i++){		
private_data_byte	**8**	**bslbf**
}		
}		

2.6.57 Semantic definition of fields in content labelling descriptor

metadata_application_format: The metadata_application_format is a 16-bit field, coded as defined in Table 2-82, that specifies the application responsible for defining usage, syntax and semantics of the content_reference_id record and of any other privately defined fields in this descriptor. See also 2.12.1. The value 0xFFFF indicates that the format is signalled by the value carried in the metadata_application_format_identifier field.

Table 2-82 – Metadata_application_format

Value	Description
0x0000 .. 0x000F	Reserved
0x0010	ISO 15706 (ISAN) encoded in its binary form (see Notes 1 and 3)
0x0011	ISO 15706-2 (V-ISAN) encoded in its binary form (see Notes 2 and 3)
0x0012 .. 0x00FF	Reserved
0x0100 .. 0xFFFE	User defined
0xFFFF	Defined by the metadata_application_format_identifier field
NOTE 1 – For ISAN, the content_reference_id_byte is set to binary encoding and the content_reference_id_record_length is set to 0x08. NOTE 2 – For V-ISAN, the content_reference_id_byte is set to binary encoding and the content_reference_id_record_length is set to 0x0C. NOTE 3 – For interoperability amongst metadata applications that use the metadata_application_format values of 0x0010 and 0x0011, it is recommended that the content_reference_id_flag be set to '1' and the content_time_base_indicator be set to '00'.	

metadata_application_format_identifier: The coding of this 32-bit field is fully equivalent to the coding of the format_identifier field in the registration_descriptor, as defined in 2.6.8.

NOTE – The assigned Registration Authority for the format_identifier field is SMPTE.

content_reference_id_record_flag: The content_reference_id_record_flag is a 1-bit flag that signals the presence of a content_reference_id_record in this descriptor.

content_time_base_indicator: The content_time_base_indicator (see Table 2-83) is a 4-bit field which specifies the used content time base. If the descriptor is associated with a program, then the content time base applies to all streams that are part of that program. A value of 1 indicates usage of the STC, while a value of 2 indicates usage of NPT, the Normal Play Time as defined in ISO/IEC 13818-6. The values between 8 and 15 indicate usage of a privately defined content time base. If coded with a value of 0, no content time base is defined in this descriptor. If no content time base is specified for a program or stream, then the mapping of time references in the metadata to the content is not defined in this Specification.

Table 2-83 – Content_time_base_indicator values

Value	Description
0	No content time base defined in this descriptor
1	Use of STC
2	Use of NPT
3 .. 7	Reserved
8 .. 15	Use of privately defined content time base

content_reference_id_record_length: The content_reference_id_record_length is an 8-bit field that specifies the number of content_reference_id_bytes immediately following this field. This field shall not be coded with the value '0'.

content_reference_id_byte: The content_reference_id_byte is part of a string of one or more contiguous bytes that assigns one or more reference identifications (labels) to the content to which this descriptor is associated. The format of this byte string is defined by the body indicated by the coded value in the metadata_application_format field.

content_time_base_value: The content_time_base_value is a 33-bit field that specifies a value in units of 90 kHz of the content time base indicated by the content_time_base_indicator field.

metadata_time_base_value: The metadata_time_base_value is a 33-bit field that is coded in units of 90 kHz. The field is coded with the value of the metadata time base at the instant in time in which the time base indicated by content_time_base_indicator reaches the value encoded in the content_time_base_value field. Note that the metadata time base may use any time-scale, but that its value is to be coded in units of 90 kHz. For example, if a SMPTE type of time code is used, then the number of hours, minutes, seconds and frames is expressed in the corresponding number of 90-kHz units.

contentId: The contentId is a 7-bit field that specifies the value of the content_Id field in the NPT Reference Descriptor for the applied NPT time base.

time_base_association_data_length: The time_base_association_data_length is an 8-bit field that specifies the number of reserved bytes immediately following this field. The reserved bytes can be used to carry time base association data for time bases defined in future.

private_data_byte: The private_data_byte is an 8-bit field. The private_data_bytes represent data, the format of which is defined privately. These bytes can be used to provide additional information as deemed appropriate. The use of these bytes is defined by the metadata application format.

2.6.58 Metadata pointer descriptor

The metadata pointer descriptor (see Table 2-84) points to a single metadata service and associates this metadata service with audiovisual content in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream. The metadata is associated with the content within the context of the descriptor. The context is defined by the location of the descriptor. In a transport stream, the descriptor may be located in the PMT in the descriptor loop for either the program or an elementary stream, but may also be located in tables not defined in this Specification, such as tables describing bouquets of broadcast services. The metadata may be located in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, but the same metadata may also be provided on alternative locations, such as the Internet.

The descriptor may contain location information of metadata that is not carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream; the coding of the location information is metadata application format specific. The descriptor allows for carriage of private data.

For metadata carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the descriptor specifies the tools used for such carriage. If the metadata is carried in PES packets, metadata sections, or ISO/IEC 13818-6 synchronized download sections, the metadata_service_id field identifies the metadata service in the referenced metadata stream. If an ISO/IEC 13818-6 carousel is used to carry the metadata, then the private data may provide information to signal the metadata service, such as the applied value of the module_id for carriage of the metadata in a data carousel, and the file name of the metadata when the object carousel is used.

Receivers should be aware that multiple metadata services may be pointed to from the same program or audiovisual stream (as defined by the context of the descriptor). A unique metadata pointer descriptor shall be used to point to each metadata service used by the program or audiovisual stream. Similarly, the same metadata service can be pointed to from several programs or audiovisual streams by using a separate metadata pointer descriptors for each association.

Table 2-84 – Metadata pointer descriptor

Syntax	No. of bits	Mnemonic
Metadata_pointer_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
metadata_application_format	**16**	**uimsbf**
if (metadata_application_format== 0xFFFF){		
metadata_application_format_identifier	**32**	**uimsbf**
}		
metadata_format	**8**	**uimsbf**
if (metadata_format== 0xFF){		
metadata_format_identifier	**32**	**uimsbf**
}		
metadata_service_id	**8**	**uimsbf**
metadata_locator_record_flag	**1**	**bslbf**
MPEG_carriage_flags	**2**	**uimsbf**
reserved	**5**	**bslbf**
if (metadata_locator_record_flag == '1'){		
metadata_locator_record_length	**8**	**uimsbf**
for (i = 0; i < metadata_locator_record_length; i++){		
metadata_locator_record_byte	**8**	**bslbf**
}		
}		
if (MPEG_carriage_flags ≤ 2){		
program_number	**16**	**uimsbf**
}		
if (MPEG_carriage_flags == 1){		
transport_stream_location	**16**	**uimsbf**
transport_stream_id	**16**	**uimsbf**
}		
for (i=0; i<N;i++){		
private_data_byte	**8**	**bslbf**
}		
}		

2.6.59 Semantic definition of fields in metadata pointer descriptor

metadata_application_format: The metadata_application_format is a 16-bit field that specifies the application responsible for defining usage, syntax and semantics of the metadata_locator_record record and any other privately defined fields in this descriptor. The coding of this field is defined in Table 2-82 in 2.6.57.

metadata_application_format_identifier: The coding of this field is defined in 2.6.57.

metadata_format: The metadata_format is an 8-bit field that indicates the format and coding of the metadata. The coding of this field is specified in Table 2-85.

Table 2-85 – Metadata format values

Value	Description
0x00 .. 0x0F	Reserved
0x10	ISO/IEC 15938-1 TeM
0x11	ISO/IEC 15938-1 BiM
0x12 .. 0x3E	Reserved
0x3F	Defined by metadata application format
0x40 .. 0xFE	Private use
0xFF	Defined by metadata_format_identifier field

The values 0x10 and 0x11 identify ISO/IEC 15938-1 defined data. The value 0x3F indicates that the format is defined by the body indicated by the metadata_application_format field. The values in the inclusive range of 0x40 up to 0xFE are available to signal use of private formats. The value 0xFF indicates that the format is signalled by the metadata_format_identifier field.

metadata_format_identifier: The coding of this 32-bit field is fully equivalent to the coding of the format_identifier field in the registration_descriptor, as defined in 2.6.8.

NOTE – SMPTE is assigned as Registration Authority for the format_identifier field.

metadata_service_id: This 8-bit field references the metadata service. It is used for retrieving a metadata service from within a metadata stream.

metadata_locator_record_flag: The metadata_locator_record_flag is a 1-bit field which, when set to '1' indicates that associated metadata is available on a location outside of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, specified in a metadata_locator_record.

MPEG_carriage_flags: The MPEG_carriage_flags is a 2-bit field which specifies if the metadata stream containing the associated metadata service is carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, and if so, whether the associated metadata is carried in a transport stream or program stream. The coding of this field is defined in Table 2-86.

Table 2-86 – MPEG_carriage_flags

Value	Description
0	Carriage in the same transport stream where this metadata pointer descriptor is carried.
1	Carriage in a different transport stream from where this metadata pointer descriptor is carried.
2	Carriage in a program stream. This may or may not be the same program stream in which this metadata pointer descriptor is carried.
3	None of the above.

metadata_locator_record_length: The metadata_locator_record_length is an 8-bit field that specifies the number of metadata_locator_record_bytes immediately following. This field shall not be coded with the value 0.

metadata_locator_record_byte: The metadata_locator_record_byte is part of a string of one or more contiguous bytes that form the metadata locator record. This record specifies one or more locations outside of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream. The format of the metadata locator record is defined by the metadata application signalled by the metadata_application_format field. The record may for example contain Internet URLs that specify where the metadata can be found, possibly in addition to their location(s) in the transport stream. If the MPEG_carriage_flags is coded with the value 0, 1 or 2 and the metadata locator record is present, then this signals alternative locations for the same metadata.

program_number: The program_number is a 16-bit field that identifies the program_number of the MPEG-2 program in the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream in which associated metadata is carried. If the MPEG_carriage_flags have the value 0, then the transport stream is the current one, and if the MPEG_carriage_flags have the value 1, it is the transport stream signalled by the field's transport_stream_location and transport_stream_id.

transport_stream_location: The transport_stream_location is a 16-bit field that is defined privately. For example, this field may be used by applications to signal the original_network_id defined by ETSI.

transport_stream_id: The transport_stream_id is a 16-bit field that identifies the transport stream in which associated metadata is carried.

private_data_byte: The private_data_byte is an 8-bit field. The private_data_bytes represent data, the format of which is defined privately. These bytes can be used to provide additional information as deemed appropriate.

2.6.60 Metadata descriptor

The metadata descriptor (see Table 2-87) specifies parameters of a metadata service carried in an MPEG-2 TS or PS. In an MPEG-2 TS, the descriptor is included in the PMT in the descriptor loop for the elementary stream that carries the metadata service. The descriptor specifies the format of the associated metadata, and contains the value of the metadata_service_id to identify the metadata service to which the metadata descriptor applies. As needed, the descriptor can convey information to identify the metadata service from a collection of metadata transmitted in a DSM-CC carousel. Optionally metadata application format specific private data can be carried.

The metadata descriptor also signals whether decoder configuration is required and is able to carry the decoder configuration bytes, but this is only practical if the number of these bytes is small. If the decoder configuration information is too large to be carried by the descriptor, it shall be contained in a metadata service. This may be within the metadata service itself, or in another metadata service within the same program. Identification of the metadata service that contains the decoder configuration is provided by the metadata descriptor. If a DSM-CC carousel is used to carry the decoder configuration, then information can be provided how to retrieve the decoder configuration from the carousel.

Table 2-87 – Metadata descriptor

Syntax	No. of bits	Mnemonic
Metadata_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
metadata_application_format	**16**	**uimsbf**
if (metadata_application_format == 0xFFFF) {		
metadata_application_format_identifier	**32**	**uimsbf**
}		
metadata_format	**8**	**uimsbf**
if (metadata_format== 0xFF){		
metadata_format_identifier	**32**	**uimsbf**
}		
metadata_service_id	**8**	**uimsbf**
decoder_config_flags	**3**	**bslbf**
DSM-CC_flag	**1**	**bslbf**
reserved	**4**	**bslbf**
if (DSM-CC_flag == '1'){		
service_identification_length	**8**	**uimsbf**
for(i=0; i<service_identification_length; i++) {		
service_identification_record_byte	**8**	**bslbf**
}		
}		
if (decoder_config_flags == '001') {		
decoder_config_length	**8**	**uimsbf**
for(i=0; i<decoder_config_length; i++) {		
decoder_config_byte	**8**	**bslbf**
}		
}		
if (decoder_config_flags == '011') {		
dec_config_identification_record_length	**8**	**uimsbf**
for(i=0;i<dec_config_id_record_length;i++) {		
dec_config_identification_record_byte	**8**	**bslbf**
}		
}		
if (decoder_config_flags == '100') {		
decoder_config_metadata_service_id	**8**	**uimsbf**
}		
if (decoder_config_flags == '101' || decoder_config_flags == '110') {		
reserved_data_length	**8**	**uimsbf**
for(i=0;i<reserved_data_length;i++) {		
reserved	**8**	**bslbf**
}		
}		
for (i=0; i<N;i++) {		
private_data_byte	**8**	**bslbf**
}		
}		

2.6.61 Semantic definition of fields in metadata descriptor

metadata_application_format: The metadata_application_format is a 16-bit field that specifies the application responsible for defining usage, syntax and semantics of the service_identification_record and any privately defined bytes in this descriptor. The coding of this field is defined in Table 2-82.

metadata_application_format_identifier: The coding of this field is defined in 2.6.57.

metadata_format: The coding of this field is defined in 2.6.59.

metadata_format_identifier: The coding of this field is defined in 2.6.59.

metadata_service_id. This 8-bit field identifies the metadata service to which this metadata descriptor applies.

decoder_config_flags: The decoder_config_flags is a 3-bit field which indicates whether and how decoder configuration information is conveyed. See Table 2-88.

Table 2-88 – decoder_config_flags

Value	Description
'000'	No decoder configuration is needed.
'001'	The decoder configuration is carried in this descriptor in the decoder_config_byte field.
'010'	The decoder configuration is carried in the same metadata service as to which this metadata descriptor applies.
'011'	The decoder configuration is carried in a DSM-CC carousel. This value shall only be used if the metadata service to which this descriptor applies is using the same type of DSM-CC carousel.
'100'	The decoder configuration is carried in another metadata service within the same program, as identified by the decoder_config_metadata_service_id field in this metadata descriptor.
'101' .. '110'	Reserved.
'111'	Privately defined.

DSM-CC_flag: This is a one-bit flag that is set to '1' if the stream with which this descriptor is associated is carried in an ISO/IEC 13818-6 data or object carousel.

NOTE 1 – The use of the object or data carousel is indicated by the applied stream-type value for this metadata stream.

service_identification length: This field specifies the number of service_identification_record_bytes immediately following.

service_identification_record_byte: This byte is part of a string of one or more contiguous bytes that specify the service_identification_record. This record contains data on retrieval of the metadata service from a DSM-CC carousel. The format of the metadata locator record is defined by the application indicated by the metadata application format. When a DSM-CC object carousel is used, the record may for example comprise the unique object identifier (the IOP:IOR() from 11.3.1 and 5.7.2.3 of ISO/IEC 13818-6 DSM-CC) for the metadata service. Similarly, in case of a DSM-CC data carousel, the record can for example provide the transaction_id and the module_id of the metadata service.

decoder_config_length: This field specifies the number of decoder_config_bytes immediately following.

decoder_config_byte: These bytes comprise the decoder configuration information. This sequence of bytes comprises the configuration information needed by the receiver to decode this service. It is intended that carriage in the metadata descriptor is only used when the configuration information is very small.

dec_config_identification_record_length: This field specifies the immediately following number of dec_config_identification_record_bytes.

dec_config_identification_record_byte: This byte is part of a string of one or more contiguous bytes that specify the dec_config_identification_record. This record specifies how to retrieve the required decoder configuration from a DSM-CC carousel. The format of the metadata locator record is defined by the metadata application format. When a DSM-CC object carousel is used, the record may for example comprise the unique object identifier (the IOP:IOR() from 11.3.1 and 5.7.2.3 of ISO/IEC 13818-6 DSM-CC) for the decoder configuration. Similarly, in case of a DSM-CC data carousel, the record may for example provide the transaction_id and the module_id of the decoder configuration.

decoder_config_metadata_service_id: This is the value of the metadata_service_id that is assigned to the metadata service that contains the decoder configuration. The metadata service indicated by the decoder_config_metadata_service_id and the metadata service that uses that decoder configuration shall be in the same program. Hence in a transport stream, the metadata descriptors for both these metadata services shall be in the same PMT. The metadata descriptor of the metadata service indicated by the decoder_config_metadata_service_id shall have a decoder_config_flag field with a value of either '001', '010' or '011'.

reserved_data_length: This field specifies the number of reserved bytes immediately following.

private_data_byte: The private_data_byte is an 8-bit field. The private_data_bytes represent data, the format of which is defined privately. These bytes can be used to provide additional information as deemed appropriate.

2.6.62 Metadata STD descriptor

This descriptor defines parameters of the STD model (defined in 2.12.10) for the processing of the metadata stream to which this descriptor is associated. See Table 2-89.

Table 2-89 – Metadata STD descriptor

Syntax	No. of bits	Mnemonic
Metadata_STD_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
reserved	**2**	**bslbf**
metadata_input_leak_rate	**22**	**uimsbf**
reserved	**2**	**bslbf**
metadata_buffer_size	**22**	**uimsbf**
reserved	**2**	**bslbf**
metadata_output_leak_rate	**22**	**uimsbf**
}		

2.6.63 Semantic definition of fields in metadata STD descriptor

metadata_input_leak_rate: The metadata_input_leak_rate is a 22-bit field that specifies the leak rate for the associated metadata stream in the T-STD model out of the buffer TB_n into buffer B_n. The leak rate is specified in units of 400 bits/s. For metadata carried in a program stream, the coding of the metadata_input_leak_rate field is not specified, as the rate into B_n equals the rate of the program stream.

metadata_buffer_size: The metadata_buffer_size is a 22-bit field that specifies the size of buffer B_n in the STD model for the associated metadata stream. The size of B_n is specified in units of 1024 bytes.

metadata_output_leak_rate: The metadata_output_leak_rate is a 22-bit field that specifies for the associated metadata service the leak rate in the STD model out of buffer B_n to the decoder. The leak rate is specified in units of 400 bits/s. For metadata streams transported synchronously (stream-type 0x15 or 0x19), the metadata access units are instantaneously removed from B_n under the control of PTS timestamps and in that case the coding of the metadata_output_leak_rate field is not specified.

2.6.64 AVC video descriptor

For AVC video streams, the AVC video descriptor provides basic information for identifying coding parameters of the associated AVC video stream, such as on profile and level parameters included in the SPS of an AVC video stream or in the subset SPS of an SVC video sub-bitstream.

For AVC video streams conforming to one or more profiles defined in Annex G, or Annex H or Annex I of Rec. ITU-T H.264 | ISO/IEC 14496 10, there may be one AVC video descriptor associated to each of the video sub-bitstreams, or MVC video subsets or MVCD video subsets identifying coding parameters of the associated re-assembled AVC video streams.

The AVC video descriptor also signals the presence of AVC still pictures, AVC 24-hour pictures as well as 3D rendering assistance SEIs such as frame packing arrangement SEI message or stereo video information SEI message in the AVC video stream. If this descriptor is not included in the PMT for an AVC video stream, a video sub-bitstream, or an MVC video sub-bitstream or an MVCD video sub-bitstream in a transport stream or in the PSM, if present, for an AVC video stream, a video sub bitstream or an MVC video sub-bitstream in a program stream, then such AVC video stream shall not contain AVC still pictures, shall not contain AVC 24-hour pictures and may or may not contain frame packing arrangement SEI message or stereo video information SEI message. (See Table 2-90.)

Table 2-90 – AVC video descriptor

Syntax	No. of bits	Mnemonic
AVC_video_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
profile_idc	**8**	**uimsbf**
constraint_set0_flag	**1**	**bslbf**
constraint_set1_flag	**1**	**bslbf**
constraint_set2_flag	**1**	**bslbf**
constraint_set3_flag	**1**	**bslbf**
constraint_set4_flag	**1**	**bslbf**
constraint_set5_flag	**1**	**bslbf**
AVC_compatible_flags	**2**	**bslbf**
level_idc	**8**	**uimsbf**
AVC_still_present	**1**	**bslbf**
AVC_24_hour_picture_flag	**1**	**bslbf**

Syntax	No. of bits	Mnemonic
Frame_Packing_SEI_not_present_flag	**1**	**bslbf**
reserved	**5**	**bslbf**
}		

2.6.65 Semantic definition of fields in AVC video descriptor

profile_idc, constraint_set0_flag, constraint_set1_flag, constraint_set2_flag, constraint_set3_flag, constraint_set4_flag, constraint_set5_flag and AVC_compatible_flags and level_idc – These fields, with the exception of AVC_compatible_flags, shall be coded according to the semantics for these fields defined in Rec. ITU-T H.264 | ISO/IEC 14496-10. The semantics of AVC_compatible_flags are exactly equal to the semantics of the field(s) defined for the 2 bits between the *constraint_set5 flag* and the *level_idc* field in the sequence parameter set, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10. The entire AVC video stream to which the AVC descriptor is associated shall conform to the profile, level and constraints signalled by these fields.

NOTE – In one or more sequences in the AVC video stream the level may be lower than the level signalled in the AVC video descriptor, while also a profile may occur that is a subset of the profile signalled in the AVC video descriptor. However, in the entire AVC video stream, only tools shall be used that are included in the profile signalled in the AVC video descriptor, if present. For example, if the main profile is signalled, then the baseline profile may be used in some sequences, but only using those tools that are in the main profile. If the sequence parameter sets in an AVC video stream signal different profiles, and no additional constraints are signalled, then the stream may need examination to determine which profile, if any, the entire stream conforms to. If an AVC video descriptor is to be associated with an AVC video stream that does not conform to a single profile, then the AVC video stream must be partitioned into two or more sub-streams, so that AVC video descriptors can signal a single profile for each such sub-stream.

AVC_still_present – This 1-bit field when set to '1' indicates that the AVC video stream may include AVC still pictures. When set to '0', then the associated AVC video stream shall not contain AVC still pictures.

AVC_24_hour_picture_flag – This 1-bit flag when set to '1' indicates that the associated AVC video stream may contain AVC 24-hour pictures. For the definition of an AVC 24-hour picture, see 2.1.2. If this flag is set to '0', the associated AVC video stream shall not contain any AVC 24-hour picture.

Frame_Packing_SEI_not_present_flag – If this flag is set to '0', then the AVC video stream shall contain either the frame packing arrangement SEI message or stereo video information SEI message. If the AVC video descriptor is present and this flag is set to '1', then the presence of either of these SEI messages is unspecified.

2.6.66 AVC timing and HRD descriptor

The AVC timing and HRD descriptor provides timing and HRD parameters of the associated AVC video stream. For each AVC video stream and for each video sub-bitstream or MVC video sub-bitstream or MVCD video sub-bitstream carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the AVC timing and HRD descriptor shall be included in the PMT or in the PSM, if PSM is present in the program stream, unless the AVC video stream, the video sub-bitstream or the MVC video sub-bitstream or MVCD video sub-bitstream carries VUI parameters with the timing_info_present_flag set to '1':

- for each IDR picture or re-assembled IDR picture; and
- for each picture or re-assembled picture that is associated with a recovery point SEI message.

Absence of the AVC timing and HRD descriptor in the PMT for an AVC video stream or a re-assembled AVC video stream signals usage of the leak method in the T-STD for the transfer from MB_n to EB_n as defined:

- in 2.14.3.1 for an AVC video stream conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10;
- in 2.14.3.5 for video sub-bitstreams of an AVC video stream conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10;
- in 2.14.3.7 for MVC video sub-bitstreams or MVCD video sub-bitstreams of an AVC video stream conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10.

But such usage can also be signalled by the hrd_management_valid_flag set to '0' in the AVC timing and HRD descriptor. If the transfer rate into buffer EB_n can be determined from HRD parameters contained in an AVC video stream or an AVC video stream re-assembled from video sub-bitstreams or MVC video sub-bitstreams or MVCD video sub-bitstreams, and if this transfer rate is used in the T-STD for the transfer between MB_n to EB_n, then the AVC timing and HRD descriptor with the hrd_management_valid_flag set to '1' shall be included in the PMT for that AVC video stream or for the re-assembled AVC video stream. (See Table 2-91.)

Table 2-91 – AVC timing and HRD descriptor

Syntax	No. of bits	Mnemonic
AVC_timing_and_HRD_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
hrd_management_valid_flag	**1**	**bslbf**
reserved	**6**	**bslbf**
picture_and_timing_info_present	**1**	**bslbf**
if (picture_and_timing_info_present) {		
90kHz_flag	**1**	**bslbf**
reserved	**7**	**bslbf**
if (90kHz_flag = = '0') {		
N	**32**	**uimsbf**
K	**32**	**uimsbf**
}		
num_units_in_tick	**32**	**uimsbf**
}		
fixed_frame_rate_flag	**1**	**bslbf**
temporal_poc_flag	**1**	**bslbf**
picture_to_display_conversion_flag	**1**	**bslbf**
reserved	**5**	**bslbf**
}		

2.6.67 Semantic definition of fields in AVC timing and HRD descriptor

hrd_management_valid_flag – This 1-bit field is only defined for use in transport streams.

When the AVC timing and HRD descriptor is associated with an AVC video stream or a re-assembled AVC video stream carried in a transport stream, then the following applies. If the hrd_management_valid_flag is set to '1', then Buffering Period SEI and Picture Timing SEI messages, as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10, shall be present in the associated AVC video stream or re-assembled AVC video stream. These Buffering Period SEI messages shall carry coded initial_cpb_removal_delay and initial_cpb_removal_delay_offset values for the NAL HRD. If the hrd_management_valid_flag is set to '1', then the transfer of each byte from MB_n to EB_n in the T-STD shall be according to the delivery schedule for that byte into the CPB in the NAL HRD, as determined from the coded initial_cpb_removal_delay and initial_cpb_removal_delay_offset values for SchedSelIdx = cpb_cnt_minus1. When the hrd_management_valid_flag is set to '0', the leak method for the transfer from MB_n to EB_n in the T-STD shall be used:

- as defined in 2.14.3.1 for AVC video streams conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10;
- as defined in 2.14.3.5 for video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10.
- as defined in 2.14.3.7 for MVC video sub-bitstreams or MVCD video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10.

When the AVC timing and HRD descriptor is associated with an AVC video stream or a re-assembled AVC video stream carried in a program stream, then the meaning of the hrd_management_valid_flag is not defined.

picture_and_timing_info_present – This 1-bit field when set to '1' indicates that the 90kHz_flag and parameters for accurate mapping to 90-kHz system clock are included in this descriptor.

90kHz_flag, N, K – The 90kHz_flag when set to '1' indicates that the frequency of the AVC time base is 90 kHz. For an AVC video stream the frequency of the AVC time base is defined by the AVC parameter time_scale in VUI parameters, as defined in Annex E of Rec. ITU-T H.264 | ISO/IEC 14496-10. The relationship between the AVC time_scale and the STC shall be defined by the parameters N and K in this descriptor as follows.

$$time\_scale = \frac{(N \times system\_clock\_frequency)}{K}$$

where time_scale denotes the exact frequency of the AVC time base, with K larger than or equal to N.

If the 90kHz_flag is set to '1', then N equals 1 and K equals 300. If the 90kHz_flag is set to '0', then the values of N and K are provided by the coded values of the N and K fields.

NOTE 1 – This allows mapping of time expressed in units of time_scale to 90-kHz units, as needed for the calculation of PTS and DTS timestamps, for example in decoders for AVC access units for which no PTS or DTS is encoded in the PES header.

num_units_in_tick – Coded exactly in the same way as the num_units_in_tick field in VUI parameters in Annex E of Rec. ITU-T H.264 | ISO/IEC 14496-10. The information provided by this field shall apply to the entire AVC video stream to which the AVC timing and HRD descriptor is associated.

fixed_frame_rate_flag – Coded exactly in the same way as the fixed_frame_rate_flag in VUI parameters in Annex E of Rec. ITU-T H.264 | ISO/IEC 14496-10. When this flag is set to '1', it indicates that the coded frame rate is constant within the associated AVC video stream. When this flag is set to '0', no information about the frame rate of the associated AVC video stream is provided in this descriptor.

temporal_poc_flag – When the temporal_poc_flag is set to '1' and the fixed_frame_rate_flag is set to '1', then the associated AVC video stream shall carry Picture Order Count (POC) information (PicOrderCnt) whereby pictures are counted in units of $\Delta t_{fi,dpb}(n)$, where $\Delta t_{fi,dpb}(n)$ is specified in equation E-10 of Rec. ITU-T H.264 | ISO/IEC 14496-10. When the temporal_poc_flag is set to '0', no information is conveyed regarding any potential relationship between the POC information in the AVC video stream and time.

NOTE 2 – This reduces the overhead necessary to signal timing for each access unit. An effective PTS and DTS can be calculated for access units for which no explicit PTS/DTS is carried. Repetition of most recently presented field of the appropriate parity (or frame) is implied when the difference between the PTSs of the current and the next picture is greater than $2 \times \Delta t_{fi,dpb}$ (or greater than $\Delta t_{fi,dpb}$ when *frame_mbs_only_flag* is equal to 1).

picture_to_display_conversion_flag – This 1-bit field when set to '1' indicates that the associated AVC video stream may carry display information on coded pictures by providing the pic_struct field in picture_timing SEI messages (see Annex D of Rec. ITU-T H.264 | ISO/IEC 14496-10) and/or by providing the Picture Order Count (POC) information (PicOrderCnt), whereby pictures are counted in units of $\Delta t_{fi,dpb}(n)$ (see also the semantics of temporal_poc_flag), so that timing information for a successive AVC access unit can be derived from the previous picture in decoding or presentation order.

When the picture_to_display_conversion_mode_flag is set to '0', then picture timing SEI messages in the AVC video stream, if present, shall not contain the pic_struct field, and hence the pic_struct_present_flag shall be set to '0' in the VUI parameters in the AVC video stream.

2.6.68 MPEG-2 AAC audio descriptor

For individual ISO/IEC 13818-7 streams directly carried in PES packets, the MPEG-2 AAC audio descriptor defined in Table 2-92 provides basic information for identifying the coding parameters of such audio elementary streams.

Table 2-92 – MPEG-2 AAC_audio_descriptor

Syntax	No. of bits	Mnemonic
MPEG-2_AAC_audio_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
MPEG-2_AAC_profile	**8**	**uimsbf**
MPEG-2_AAC_channel_configuration	**8**	**uimsbf**
MPEG-2_AAC_additional_information	**8**	**uimsbf**
}		

2.6.69 Semantic definition of fields in MPEG-2 AAC audio descriptor

MPEG-2_AAC_profile – This 8-bit field indicates the AAC profile according to the index in Table 31 of ISO/IEC 13818-7:2006.

MPEG-2_AAC_channel_configuration – This 8-bit field indicates the number and configuration of audio channels presented to the listener by the AAC decoder for the specified program. Values in the range from 1 to 6 indicate number and configuration of audio channels as given for "Default bitstream index number" in Table 42 of ISO/IEC 13818-7:2006. All other values indicate that the number and configuration of audio channels is undefined.

MPEG-2_AAC_additional_information – This 8-bit field indicates whether or not bandwidth extension data as defined in ISO/IEC 13818-7:2006 is embedded in the AAC bitstream according to Table 2-93.

Table 2-93 – MPEG-2_AAC_additional_information field values

Value	Description
0x00	AAC data according to ISO/IEC 13818-7:2006
0x01	AAC data with Bandwidth Extension data present according to ISO/IEC 13818-7:2006
0x02 .. 0xFF	Reserved

2.6.70 MPEG-4 text descriptor

The MPEG-4 text descriptor (see Table 2-94) carries textConfig() specified in ISO/IEC 14496-17 for the associated ISO/IEC 14496-17 text stream, thereby providing basic information needed for the decoding of the associated ISO/IEC 14496-17 stream. For each ISO/IEC 14496-17 text stream carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the MPEG-4 text descriptor shall be included in the PMT or in the PSM, if PSM is present in the program stream.

Table 2-94 – MPEG-4 text descriptor

Syntax	No. of bits	Mnemonic
MPEG-4_text_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
textConfig()		
}		

2.6.71 Semantic definition of fields in MPEG-4 text descriptor

textConfig() – This shall carry the TextConfig() of the associated ISO/IEC 14496-17 text stream, as defined in ISO/IEC 14496-17.

2.6.72 MPEG-4 audio extension descriptor

The MPEG-4 audio extension descriptor (see Table 2-95) carries zero or more audioProfileLevelIndication fields and zero or one audioSpecificConfig() field, both encoded as specified in ISO/IEC 14496-3. Note that for each ISO/IEC 14496-3 audio stream carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, it is required that the MPEG-4 audio descriptor be included in the PMT or in the PSM, if PSM is present in the program stream, while the MPEG-4 audio extension descriptor may be present too, providing additional information. If in the MPEG-4 audio descriptor the MPEG-4_audio_profile_and_level field is encoded with the value 0xFF, indicating that the audio profile and level is not specified in the MPEG-4 audio descriptor, then the MPEG-4 audio extension descriptor shall be present in the same PMT or PSM as the MPEG-4 audio descriptor. Note that this descriptor allows to provide the audioSpecificConfig out of band, so as to allow receivers to retrieve information about the associated audio stream without accessing the stream itself. The descriptor also allows to associate an audioSpecificConfig to an audio stream.

Table 2-95 – MPEG-4 audio extension descriptor

Syntax	No. of bits	Mnemonic
MPEG-4_audio_extension_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
ASC_flag	**1**	**bslbf**
reserved	**3**	**bslbf**
num_of_loops	**4**	**uimsbf**
for (i=0; i<num_of_loops; i++) {		
audioProfileLevelIndication	**8**	**uimsbf**
}		
if (ASC_flag == '1') {		
ASC_size	**8**	**uimsbf**
audioSpecificConfig()		
}		
}		

2.6.73 Semantic definition of fields in MPEG-4 audio extension descriptor

ASC_flag – A one-bit flag signalling the presence of the ASC_size field in this descriptor.

num_of_loops – A 4-bit field specifying the number of immediately following audioprofileLevelIndication fields in this descriptor. This field may be encoded with the value zero.

audioProfileLevelIndication – The audio profile and level of the associated ISO/IEC 14496-3 audio stream, encoded as specified for the audioprofileLevelIndication field in 1.5.2.1 in ISO/IEC 14496-3. Note that a single ISO/IEC 14496-3 audio stream may comply to more than one audio profile and level, and that this descriptor is designed to convey up to 15 different audioprofileLevelIndication values.

ASC_size – The number of bytes of the immediately following AudioSpecificConfig().

audioSpecificConfig() – The audioSpecificConfig() of the associated ISO/IEC 14496-3 audio stream, as specified in 1.6.2.1 in ISO/IEC 14496-3.

2.6.74 Auxiliary video stream descriptor

The auxiliary video stream descriptor (see Table 2-96) specifies parameters for the decoding and interpretation of the auxiliary video stream to which the descriptor is associated. For each auxiliary video stream carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the auxiliary video stream descriptor shall be included in the PMT or in the PSM, if PSM is present in the program stream.

Table 2-96 – Auxiliary video stream descriptor

Syntax	No. of bits	Mnemonic
Auxiliary_video_stream_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
aux_video_codedstreamtype	**8**	**uimsbf**
si_rbsp(descriptor_length-1)		
}		

2.6.75 Semantic definition of fields in auxiliary video stream descriptor

aux_video_codedstreamtype – An 8-bit unsigned integer that indicates the compression coding type of the auxiliary video stream. The value of aux_video_codedstreamtype shall match one of the stream types defined in Table 2-34 for video (for instance 0x02, 0x10 or 0x1B). In order to convey additional information such as profile/level, a descriptor that corresponds to the aux_video_codedstreamtype may also be included in the PMT or in the PSM, if PSM is present in the program stream, for the auxiliary video data stream.

NOTE – For example, if the auxiliary video is encoded using Rec. ITU-T H.264 | ISO/IEC 14496-10 Video, then the value of aux_video_codedstreamtype is 0x1B and an AVC video descriptor (descriptor_tag = 40) can be optionally included.

si_rbsp() – Supplemental information RBSP as defined in ISO/IEC 23002-3. It shall contain at least one auxiliary video supplemental information (AVSI) message (also defined in ISO/IEC 23002-3). The type of auxiliary video is inferred from si_rbsp(). The total size of si_rbsp() shall not exceed 254 bytes.

2.6.76 SVC extension descriptor

For video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, the SVC extension descriptor (see Table 2-97) provides information about the AVC video stream resulting from re-assembling (up to) the associated video sub-bitstream and provides information about scalability and re-assembly of the associated video sub-bitstream. There may be one SVC extension descriptor associated with any of the video sub-bitstreams of an AVC video stream conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10.

Table 2-97 – SVC extension descriptor

Syntax	No. of bits	Mnemonic
SVC_extension_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
width	**16**	**uimsbf**
height	**16**	**uimsbf**
frame_rate	**16**	**uimsbf**
average_bitrate	**16**	**uimsbf**
maximum_bitrate	**16**	**uimsbf**
dependency_id	**3**	**bslbf**
reserved	**5**	**bslbf**
quality_id_start	**4**	**bslbf**
quality_id_end	**4**	**bslbf**
temporal_id_start	**3**	**bslbf**
temporal_id_end	**3**	**bslbf**
no_sei_nal_unit_present	**1**	**bslbf**
reserved	**1**	**bslbf**
}		

2.6.77 Semantic definition of fields in SVC extension descriptor

width – This 16-bit field indicates the maximum image width resolution, in pixels of the re-assembled AVC video stream.

height – This 16-bit field indicates the maximum image height resolution, in pixels of the re-assembled AVC video stream.

frame_rate – This 16-bit field indicates the maximum frame rate, in frames/256 seconds of the re-assembled AVC video stream.

average_bitrate – This 16-bit field indicates the average bit rate, in kbit per second, of the re-assembled AVC video stream.

maximum_bitrate – This 16-bit field indicates the maximum bit rate, in kbit per second, of the re-assembled AVC video stream.

dependency_id – This 3-bit field indicates the value of dependency_id associated with the video sub-bitstream.

quality_id_start – This 4-bit field indicates the minimum value of the quality_id of the NAL unit header syntax element of all the NAL units contained in the associated video sub-bitstream.

quality_id_end – This 4-bit field indicates the maximum value of the quality_id of the NAL unit header syntax element of all the NAL units contained in the associated video sub-bitstream.

temporal_id_start – This 3-bit field indicates the minimum value of the temporal_id of the NAL unit header syntax element of all the NAL units contained in the associated video sub-bitstream.

temporal_id_end – This 3-bit field indicates the maximum value of the temporal_id of the NAL unit header syntax element of all the NAL units contained in the associated video sub-bitstream.

no_sei_nal_unit_present – This 1-bit flag when set to '1' indicates that no SEI NAL units are present in the associated video sub-bitstream.

NOTE – In case the no_sei_nal_unit_present flag is set to '1' for all SVC video sub-bitstreams and is not set to '1' or not present for the AVC video sub-bitstream of SVC, any SEI NAL units, if present, are included in the AVC video sub-bitstream of SVC. If the SVC extension descriptor is absent for all video sub-bitstreams, SEI NAL units may be present in any SVC dependency representation of an SVC video sub-bitstream, and may require re-ordering to the order of NAL units within an access unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 before access unit re-assembling.

2.6.78 MVC extension descriptor

For MVC video sub-bitstreams of AVC video streams conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, the MVC extension descriptor (see Table 2-98) provides information about the AVC video stream resulting from reassembling (up to) the associated MVC video sub-bitstream and provides information about the contained MVC video sub-bitstream and for the reassembly of the associated MVC video sub-bitstream. There may be one MVC extension descriptor associated with any of the MVC video sub-bitstreams (with stream_type equal to 0x20) of an AVC video stream conforming to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10. When the MVC video sub-bitstream is an MVC base view sub-bitstream, the MVC extension descriptor shall be present in the associated PMT or PSM for stream_type equal to 0x1B.

This descriptor can also be used by applications that require association between stereoscopic MVC views and left or right eye using the syntax elements 'view_association_not_present' and 'base_view_is_left_eyeview'.

Table 2-98 – MVC extension descriptor

Syntax	No. of bits	Mnemonic
MVC_extension_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
average_bit_rate	**16**	**uimsbf**
maximum_bitrate	**16**	**uimsbf**
view_association_not_present	**1**	**bslbf**
base_view_is_left_eyeview	**1**	**bslbf**
reserved	**2**	**bslbf**
view_order_index_min	**10**	**bslbf**
view_order_index_max	**10**	**bslbf**
temporal_id_start	**3**	**bslbf**
temporal_id_end	**3**	**bslbf**
no_sei_nal_unit_present	**1**	**bslbf**
no_prefix_nal_unit_present	**1**	**bslbf**
}		

2.6.79 Semantics of fields in MVC extension descriptor

average_bitrate – This 16-bit field indicates the average bit rate, in kbits per second, of the re-assembled AVC video stream. When set to 0, the average bit rate is not indicated.

maximum_bitrate – This 16-bit field indicates the maximum bit rate, in kbits per second, of the re-assembled AVC video stream. When set to 0, the maximum bit rate is not indicated.

view_association_not_present – This 1-bit flag when set to '0' indicates that the syntax element base_view_is_left_eyeview signals the association between base view and left or right eye. When this flag is set to '1' no such association is signalled.

base_view_is_left_eyeview – This flag shall be set to '1' when the view_association_not_present_flag is set to '1' and no view association is conveyed in the descriptor. When the view_association_not_present_flag is set to '0' and this flag is set to '1', it indicates that the base view is associated with the left eye view (or enhancement view is associated with the right eye view). When the view_association_not_present_flag is set to '0' and this flag is set to '0', it indicates that the base view is associated with the right eye view (or enhancement view is associated with the left eye view).

view_order_index_min – This 10-bit field indicates the minimum value of the view order index of all the NAL units contained in the associated MVC video sub-bitstream.

view_order_index_max – This 10-bit field indicates the maximum value of the view order index of all the NAL units contained in the associated MVC video sub-bitstream.

temporal_id_start – This 3-bit field indicates the minimum value of the temporal_id of the NAL unit header syntax element of all the NAL units contained in the associated MVC video sub-bitstream.

temporal_id_end – This 3-bit field indicates the maximum value of the temporal_id of the NAL unit header syntax element of all the NAL units contained in the associated MVC video sub-bitstream.

no_sei_nal_unit_present – This 1-bit flag when set to '1' indicates that no SEI NAL units are present in the associated video sub–bitstream.

NOTE – In case the no_sei_nal_unit_present flag is set to '1' for all MVC video sub-bitstreams and is not set to '1' or not present for the AVC video sub-bitstream of MVC, any SEI NAL units, if present, are included in the AVC video sub-bitstream of MVC. If the MVC extension descriptor is absent for all MVC video sub-bitstreams, SEI NAL units may be present in any MVC view-component subset of an MVC video sub-bitstream, and may require re-ordering to the order of NAL units within an access unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 before access unit re-assembling.

no_prefix_nal_unit_present – This 1-bit flag when set to '1' indicates that no prefix NAL units are present in either the AVC video sub-bitstream of MVC or MVC video sub-bitstreams. When this bit is set to '0', it indicates that prefix NAL units are present in the AVC video sub-bitstream of MVC only.

2.6.80 J2K video descriptor

For J2K video elementary streams conforming to one or more profiles defined in Rec. ITU-T T.800 | ISO/IEC 15444-1, the J2K video descriptor (see Table 2-99) provides information that may be present in each J2K access unit as well as for the J2K video sequence. In addition, it provides information to signal J2K still pictures. This descriptor shall be included for each J2K video elementary stream component in the PMT with stream_type equal to 0x21.

Table 2-99 – J2K video descriptor

Syntax	No. of bits	Mnemonic
J2K_video_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
profile_and_level	**16**	**uimsbf**
horizontal_size	**32**	**uimsbf**
vertical_size	**32**	**uimsbf**
max_bit_rate	**32**	**uimsbf**
max_buffer_size	**32**	**uimsbf**
DEN_frame_rate	**16**	**uimsbf**
NUM_frame_rate	**16**	**uimsbf**
color_specification	**8**	**bslbf**
still_mode	**1**	**bslbf**
interlaced_video	**1**	**bslbf**
reserved	**6**	**bslbf**
for (i=0; i < N; i++) {		
private_data_byte	**8**	**bslbf**
}		
}		

2.6.81 Semantics of fields in J2K video descriptor

profile_and_level – This field shall be in the range 0x0101-0x04ff and coded as defined in Table A.10 of Rec. ITU-T T.800 | ISO/IEC 15444-1 and indicates broadcast profile and level values.

horizontal_size – This field shall be coded the same as Xsiz parameter found in the J2K codestream main header, as defined in Annex A of Rec. ITU-T T.800 | ISO/IEC 15444-1.

vertical_size – This field shall be coded the same as Ysiz parameter found in the J2K codestream main header, as defined in Annex A of Rec. ITU-T T.800 | ISO/IEC 15444-1.

max_bit_rate – This field may be coded the same as the Maxbr value in the j2k_brat field box specified in Table S.1 and shall not exceed the maximum compressed bit rate value for the profile and level specified in Table S.2. This field shall be set appropriately and signalled when profile_and_level = 0x0307, where no maximum bit rate is specified.

max_buffer_size – This field shall not exceed the Maximum buffer size value for the profile and level specified in the j2k_brat box in Table S.2. When profile_and_level = 0x0307, the max_buffer_size shall be set appropriately and shall not exceed (max_bit_rate/1.60E5), where max_bit_rate is expressed in bit/s.

DEN_frame_rate – This field shall be coded the same as frat_denominator field in the j2k_frat box specified in Table S.1 (see Annex S).

NUM_frame_rate – This field shall be coded the same as frat_numerator field in the frat box specified in Table S.1 (see Annex S).

NOTE – J2K frame rate is derived from the DEN_frame_rate and NUM_frame_rate values. Table 2-100 lists examples of typical broadcast frame rates with associated values of DEN_frame_rate and NUM_frame_rate.

Table 2-100 – Example frame rates based on DEN_frame_rate and NUM_frame_rate values

DEN_frame_rate	NUM_frame_rate	Frame rate ratio (decimal representation)	Frame rate
'0000 0000 0000 0000'			Forbidden
'0000 0011 1110 1001'	'0101 1101 1100 0000'	24 000 / 1001	23.976
'0000 0000 0000 0001'	'0000 0000 0001 1000'	24 / 1	24.0
'0000 0000 0000 0001'	'0000 0000 0001 1001'	25 / 1	25.0
'0000 0011 1110 1001'	'0111 0101 0011 0000'	30 000 / 1001	29.97
'0000 0000 0000 0001'	'0000 0000 0001 1110'	30 / 1	30.0
'0000 0000 0000 0001'	'0000 0000 0011 0010'	50 / 1	50.0
'0000 0011 1110 1001'	'1110 1010 0110 0000'	60 000 / 1001	59.94
'0000 0000 0000 0001'	'0000 0000 0011 1100'	60 / 1	60.00

color_specification – This field shall be coded the same as the bcol_colrc 8-bit field of the j2k_bcol box as specified in Table S.1 (see Annex S).

still_mode – This 1-bit field, when set to '1', indicates that the J2K video stream may include J2K still pictures. When set to '0', then the associated J2K video stream shall not contain J2K still pictures.

interlaced_video – This 1-bit field indicates whether the J2K video stream contains interlaced video. When this flag is set to '1' the J2K access unit elementary stream header (see Table S.1) shall include the syntax elements Auf2, fiel_box_code, fic and fio. When this flag is set to '0', these syntax elements shall not be present in the J2K access unit elementary stream header.

2.6.82 MVC operation point descriptor

The MVC operation point descriptor (see Table 2-101) provides a method to indicate profile and level for one or more operation points each constituted by a set of one or more MVC video sub-bitstreams. If present, the MVC operation point descriptor shall be included in the group of data elements following immediately the program_info_length field in the program_map_section. If an MVC operation point descriptor is present within a program description, at least one hierarchy descriptor shall be present for each MVC video sub-bitstream present in the same program.

NOTE – In order to indicate different profiles, one MVC operation point descriptor per profile is needed.

Table 2-101 – MVC operation point descriptor

Syntax	No. of bits	Mnemonic
MVC_operation_point_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
profile_idc	**8**	**uimsbf**
constraint_set0_flag	**1**	**bslbf**
constraint_set1_flag	**1**	**bslbf**
constraint_set2_flag	**1**	**bslbf**
constraint_set3_flag	**1**	**bslbf**
constraint_set4_flag	**1**	**bslbf**
constraint_set5_flag	**1**	**bslbf**
AVC_compatible_flags	**2**	**bslbf**
level_count	**8**	**uimsbf**
for (i = 0; i < level_count; i++) {		
level_idc	**8**	**uimsbf**
operation_points_count	**8**	**uimsbf**
for (j =0; j< operation_points_count; j++) {		
reserved	**5**	**bslbf**
applicable_temporal_id	**3**	**uimsbf**
num_target_output_views	**8**	**uimsbf**

Syntax	No. of bits	Mnemonic
ES_count	**8**	**uimsbf**
for (k =0; k< ES_count; k++) {		
reserved	**2**	**bslbf**
ES_reference	**6**	**uimsbf**
}		
}		
}		
}		

2.6.83 Semantic definition of fields in MVC operation point descriptor

profile_idc – This 8-bit field indicates the profile, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10, of all operation points described within this descriptor for the MVC bitstream.

constraint_set0_flag, constraint_set1_flag, constraint_set2_flag, constraint_set3_flag, constraint_set4_flag, constraint_set5_flag – These fields shall be coded according to the semantics for these fields defined in Rec. ITU-T H.264 | ISO/IEC 14496-10.

AVC_compatible_flags – The semantics of AVC_compatible_flags are exactly equal to the semantics of the field(s) defined for the 2 bits between the constraint_set2 flag and the level_idc field in the sequence parameter set, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10.

level_count – This 8-bit field indicates the number of levels for which operation points are described.

level_idc – This 8-bit field indicates the level, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10, of the MVC bitstream for the operation points described by the following groups of data elements.

operation_points_count – This 8-bit field indicates the number of operation points described by the list included in the following group of data elements.

applicable_temporal_id – This 3-bit field indicates the highest value of the temporal_id of the VCL NAL units in the re-assembled AVC video stream.

num_target_output_views – This 8-bit field indicates the value of the number of the views, targeted for output for the associated operation point.

ES_count – This 8-bit field indicates the number of ES_reference values included in the following group of data elements. The elementary streams indicated in the following group of data elements together form an operation point of the MVC video bitstream. The value 0xff is reserved.

ES_reference – This 6-bit field indicates the hierarchy layer index value present in the hierarchy descriptor which identifies a video sub-bitstream.

NOTE – The profile and level for a single operation point, e.g., the entire MVC video bitstream, can be signalled using the AVC video descriptor. Beyond that, MVC allows for decoding different view subsets which can require different profiles and/or levels. The specification of the MVC operation point descriptor supports the indication of different profiles and levels for multiple operation points.

2.6.84 MPEG2_stereoscopic_video_ format_ descriptor

The MPEG2_stereoscopic_video_format_descriptor (see Table 2-102) may be associated in the PMT for MPEG-2 video components (with stream_type value equal to 0x02). When present, the descriptor shall be located in the loop following ES_info_length field in PMT. When the descriptor is included in the PMT, the associated MPEG-2 video elementary stream shall contain stereoscopic video format information in the user_data extension as specified in Rec. ITU-T H.262 | ISO/IEC 13818-2:2000/Amd.4. If the descriptor is not included for MPEG-2 video with stream_type value equal to 0x02, then the associated MPEG-2 video elementary stream may or may not contain stereoscopic video format information.

Table 2-102 – MPEG2_stereoscopic_video_format_descriptor syntax

Syntax	No. of bits	Format
MPEG2_stereoscopic_video_format_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**

Syntax	No. of bits	Format
stereo_video_arrangement_type_present	**1**	**bslbf**
if (stereo_video_arrangement_type_present) {		
arrangement_type	**7**	**bslbf**
}		
else {		
reserved	**7**	**bslbf**
}		
}		

2.6.85 Semantic definition of fields in the MPEG2_stereoscopic_video_format_descriptor

stereo_video_arrangement_type_present: When this bit is set to '1', then the following 7-bits indicate the type of stereo_video_format_type included in the user_data of associated MPEG-2 video elementary stream. If this bit is set to '0', then no such signalling is made available in this descriptor.

arrangement_type: This field shall be set to the same value as arrangement_type defined in Table L-1 of Rec. ITU-T H.262 | ISO/IEC 13818-2:2000/Amd.4 and included in the user_data extension of associated MPEG-2 video elementary stream.

2.6.86 Stereoscopic_program_info_descriptor

Stereoscopic_program_info_descriptor (see Table 2-103) specifies the identification of 2D-only (monoscopic), frame compatible stereoscopic 3D as well as service-compatible stereoscopic 3D services. This descriptor conveys information at a program level and assists decoders determine resources required to support the signalled services. When present, this descriptor shall be included in the loop following program_info_length field in the PMT. In addition, when this descriptor is present, stream_type values and descriptors associated with the video components in the loop following ES_info_length in the PMT shall not conflict with the stereoscopic_service_type signalled in this descriptor.

Table 2-103 – Stereoscopic_program_info_descriptor syntax

Syntax	No. of bits	Format
Stereoscopic_program_info_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
reserved	**5**	**bslbf**
stereoscopic_service_type	**3**	**bslbf**
}		

2.6.87 Semantic definition of fields in the stereoscopic_program_info_descriptor

descriptor_length: This field shall be set to 0x01.

stereoscopic_service_type: This specifies the type of service that is provided through the associated components such as monoscopic, frame compatible stereoscopic or service-compatible stereoscopic according to Table 2-104.

Table 2-104 – Stereoscopic_service_type values

Values	Description
'000'	unspecified
'001'	2D-only (monoscopic) service (see Note 1)
'010'	Frame-compatible stereoscopic 3D service
'011'	Service-compatible stereoscopic 3D service (see Note 2)
'100' .. '111'	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 reserved
NOTE 1 – 2D-only (monoscopic) service is used in 2D/3D mixed programs based on service-compatible stereoscopic 3D service, i.e., a service that has an arbitrary mixture of 2D and 3D video. 2D video can be coded independently using either MPEG-2 or AVC video to maintain stability of 3DTV broadcasting system. NOTE 2 – Service-compatible stereoscopic 3D service is based on 'simulcast' of stereoscopic view sequences. Each view of the stereoscopic video sequences can be coded independently using either MPEG-2 or AVC video or any combination thereof. Base view video stream for service-compatible stereoscopic 3D services is signalled using stream_type value of 0x02 for MPEG-2 video and stream_type value of 0x1B for AVC video. Additional view video stream for service-compatible stereoscopic 3D services is signalled using stream_type value of 0x22 for MPEG-2 video and stream_type value of 0x23 for AVC video.	

2.6.88 Stereoscopic_video_info_descriptor

The stereoscopic_video_info_descriptor (see Table 2-105) provides information related to service-compatible stereoscopic 3D services that carry left and right view in separate video streams. The two streams are called the "base view video stream" and the "additional view video stream". The base view video stream may be displayed on its own for a 2D video service. If signalled as specified below, the additional view video stream may also be displayed on its own for a 2D video service.

The stereoscopic_video_info_descriptor is located in the loop following ES_info_length field in PMT. The stereoscopic_video_info_descriptor shall be included for both the base view video component (stream_type values 0x02 or 0x1B) and additional view video component (stream_type values 0x22 or 0x23) in the PMT for programs that support service-compatible stereoscopic 3D video. The stereoscopic_video_info_descriptor should not be associated for components with any other stream_type values as its meaning is undefined for other stream_type values.

Table 2-105 – Stereoscopic_video_info_descriptor syntax

Syntax	No. of bits	Format
Stereoscopic_ video_info_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
reserved	**7**	**bslbf**
base_video_flag	**1**	**bslbf**
if (base_video_flag) {		
reserved	**7**	**bslbf**
leftview_flag	**1**	**bslbf**
} else {		
reserved	**7**	**bslbf**
usable_as_2D	**1**	**bslbf**
horizontal_upsampling_factor	**4**	**bslbf**
vertical_upsampling_factor	**4**	**bslbf**
}		
}		

2.6.89 Semantic definition of fields in the stereoscopic_video_info_descriptor

base_video_flag: When the bit is set to '1', it indicates that the video stream is a base video stream. When the bit is set to '0', it indicates that the video stream is an additional view video stream. This bit shall be set to '1' for base view video stream components with stream_type values of 0x02 and 0x1B and this bit shall be set to '0' for the additional view video stream components with stream_type values of 0x22 and 0x23.

leftview_flag: When the bit is set to '1', it indicates that the associated video stream component is the left view video stream. When the bit is set to '0', it indicates that the associated video stream component is the right view video stream.

usable_as_2D: When this bit is set to '1', it indicates that the additional view video stream may also be used for a 2D video service.

horizontal_upsampling_factor and vertical_upsampling_factor: These fields provide higher level information on any upsampling that may be required after the video component is decoded. The values and description of upsampling factors are defined in Table 2-106. These values are informational, and that the definitive values are those in the video elementary stream. When this syntax element is set to '0001' decoders are expected to use the information in the video elementary stream to determine appropriate upsampling factors (if needed).

Table 2-106 – Upsampling factor values

Value	Description
'0000'	Forbidden
'0001'	unspecified
'0010'	Coded resolution is same as coded resolution of base view
'0011'	Coded resolution is ¾ coded resolution of base view
'0100'	Coded resolution is 2/3 coded resolution of base view
'0101'	Coded resolution is ½ coded resolution of base view
'0110' .. '1000'	reserved
'1001' .. '1111'	user_private

2.6.90 Extension descriptor

The extension descriptor (see Table 2-107) provides a mechanism to extend the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 descriptor range (see Table 2-45). The descriptors which are based on the extension descriptor are signalled using the extension descriptor with extension_descriptor_tag values defined in Table 2-108.

Table 2-107 – Extension descriptor

Syntax	No. of bits	Mnemonic
Extension_descriptor () {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
extension_descriptor_tag	**8**	**uimsbf**
if (extension_descriptor_tag == 0x02) {		
ObjectDescriptorUpdate()		
}		
else if (extension_descriptor_tag == 0x03) {		
HEVC_timing_and_HRD_descriptor()		
}		
else if (extension_descriptor_tag == 0x04) {		
af_extensions_descriptor ()		
}		
else if (extension_descriptor_tag == 0x05) {		
HEVC_operation_point_descriptor( )		
}		
else if (extension_descriptor_tag == 0x06) {		
HEVC_hierarchy_extension_descriptor( )		
}		
else if (extension_descriptor_tag == 0x07) {		
Green_extension_descriptor ()		
}		
else if (extension_descriptor_tag == 0x08) {		
MPEG-H_3dAudio_descriptor()		
}		
else if (extension_descriptor_tag == 0x09) {		
MPEG-H_3dAudio_config_descriptor()		
}		

Table 2-107 – Extension descriptor

Syntax	No. of bits	Mnemonic
else if (extension_descriptor_tag == 0x0A) {		
MPEG-H_3dAudio_scene_descriptor()		
}		
else if (extension_descriptor_tag == 0x0B) {		
MPEG-H_3dAudio_text_label_descriptor()		
}		
else if (extension_descriptor_tag == 0x0C) {		
MPEG-H_3dAudio_multi-stream_descriptor()		
}		
else if (extension_descriptor_tag == 0x0D) {		
MPEG-H_3dAudio_drc_loudness_descriptor()		
}		
else if (extension_descriptor_tag == 0x0E) {		
MPEG-H_3dAudio_command_descriptor()		
}		
else if (extension_descriptor_tag == 0x0F) {		
Quality_extension_descriptor ()		
}		
else if (extension_descriptor_tag == 0x10) {		
Virtual_segmentation_descriptor ()		
}		
else {		
for (i=0; i<N; i++) {		
reserved	**8**	**bslbf**
}		
}		
}		

2.6.91 Semantic definition of fields in the extension descriptor

descriptor_tag – The descriptor_tag is an 8-bit field whose value is defined in Table 2-45.

descriptor_length – The descriptor_length is an 8-bit field specifying the number of bytes of the descriptor immediately following the descriptor_length field.

extension_descriptor_tag – The extension_descriptor_tag is an 8-bit field which identifies each descriptor that uses this tag value. See Table 2-108 for the extension_descriptor_tag values.

ObjectDescriptorUpdate() – This structure is defined in section 8.5.5.2 of ISO/IEC 14496-1.

HEVC_timing_and_HRD_descriptor() – This structure is defined in 2.6.95 and 2.6.96.

af_extensions_descriptor() – This structure is defined in 2.6.99.

HEVC_operation_point_descriptor( ) – This structure is defined in 2.6.100 and 2.6.101.

HEVC_hierarchy_extension_descriptor( ) – This structure is defined in 2.6.102 and 2.6.103.

Green_extension_descriptor() – This structure is defined in 2.6.104 and 2.6.105.

MPEG-H_3dAudio_descriptor() – This structure is defined in 2.6.106 and 2.6.107.

MPEG-H_3dAudio_config_descriptor() – This structure is defined in 2.6.108 and 2.6.109.

MPEG-H_3dAudio_scene_descriptor() – This structure is defined in 2.6.110 and 2.6.111.

MPEG-H_3dAudio_text_label_descriptor() – This structure is defined in 2.6.112 and 2.6.113.

MPEG-H_3dAudio_multi-stream_descriptor() – This structure is defined in 2.6.114 and 2.6.115.

MPEG-H_3dAudio_drc_loudness_descriptor() – This structure is defined in 2.6.116 and 2.6.117.

MPEG-H_3dAudio_command_descriptor() – This structure is defined in 2.6.118.

Quality_extension_descriptor() – This structure is defined in 2.6.119.

Virtual_segmentation_descriptor() – This structure is defined in 2.6.120 and 2.6.121.

Table 2-108 – Extension descriptor tag values

Extension_descriptor_tag	TS	PS	Identification
0	n/a	n/a	Reserved
1	n/a	X	Forbidden
2	X	X	ODUpdate_descriptor
3	X	n/a	HEVC_timing_and_HRD_descriptor()
4	X	n/a	af_extensions_descriptor()
5	X	n/a	HEVC_operation_point_descriptor( )
6	X	n/a	HEVC_hierarchy_extension_descriptor( )
7	X	n/a	Green_extension_descriptor()
8	X	n/a	MPEG-H_3dAudio_descriptor()
9	X	n/a	MPEG-H_3dAudio_config_descriptor()
0x0A	X	n/a	MPEG-H_3dAudio_scene_descriptor()
0x0B	X	n/a	MPEG-H_3dAudio_text_label_descriptor()
0x0C	X	n/a	MPEG-H_3dAudio_multi-stream_descriptor()
0x0D	X	n/a	MPEG-H_3dAudio_drc_loudness_descriptor()
0x0E	X	n/a	MPEG-H_3dAudio_command_descriptor()
0x0F	X	n/a	Quality_extension_descriptor()
0x10	X	n/a	Virtual_segmentation_descriptor()
0x11 .. 0xFF	n/a	n/a	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Reserved

2.6.92 ODUpdate_descriptor

The ODUpdate_descriptor may be used to carry a set of ObjectDescriptors through an ObjectDescriptorUpdate, as a replacement or as a complement to ISO/IEC 14496 object descriptor streams defined in the IOD. If used, the ObjectDescriptorUpdate command shall be processed by the MPEG-4 terminal as defined in 7.2.5.5.2 of ISO/IEC 14496-1. The descriptors carried in the ODUpdate_descriptor are in the same name scope as the scene description described in the InitialObjectDescriptor carried in the IOD descriptor.

When an ODUpdate_descriptor is used within a transport stream, the ODUpdate_descriptor shall be conveyed in the descriptor loop immediately following the program_info_length field in the program map table, and shall be included after an IOD descriptor.

When an ODUpdate_descriptor is used within a program stream, the ODUpdate_descriptor shall be conveyed in the descriptor loop immediately following the program_stream_info_length field in the program stream map, and shall be included after an IOD descriptor.

If an ODUpdate_descriptor is included before an IOD descriptor or if IOD descriptor is not present, then the ODUpdate descriptor shall be ignored. More than one ODUpdate_descriptor may be included in a program map table or program stream map.

2.6.93 Transport_profile_descriptor

The Transport_profile_descriptor (see Table 2-109) may be associated in the PMT to signal a profile value of transport stream in the associated program. When present, the descriptor shall only be located in the loop following the program_info_length field in the PMT. If the descriptor is not included in the PMT, then the associated transport stream conforms to the complete profile.

Table 2-109 – Transport_profile_descriptor syntax

Syntax	No. of bits	Mnemonic
Transport_profile_descriptor{		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
transport_profile	**8**	**uimsbf**
for (i=0; i<N; i++) {		
private_data	**8**	**bslbf**
}		
}		

2.6.94 Semantic definition of fields in the Transport_profile_descriptor

transport_profile – This 8-bit profile value signals the use of constraints in the associated transport stream for the program. See Table 2-110.

Table 2-110 – Transport_profile values

Values	Description
0x00	unspecified
0x01	Complete profile [a]
0x02	Adaptive profile [b]
0x03 .. 0x0E	reserved
0x0F .. 0xFF	user_private [c]

[a] Transport streams using this profile conform to all the normative definitions for transport streams. These include conformant discontinuities, PCR jitter/accuracy, strict T-STD management, PCR interval conformance (less than 100 ms), as well as PTS/DTS interval (0.7 seconds) and compliance.

[b] Transport streams using this profile conform to all the normative definitions for transport streams with the following exceptions:

- The PCR jitter may exceed the specified tolerance as applications that use this profile usually do not include null-PID packets. Clients that process these streams usually do not use the PCR to derive the decoder STC. However, the PCR value can be used in conjunction with the PTS and DTS for conformant STD management of all the media components in the associated program;
- the PCR interval occasionally exceeds 100 ms in applications that use this profile due to occasional bit rate variations in certain locations;
- conforming continuity counter errors and time base discontinuity may occur more frequently than in complete profile.

[c] User private values of transport_profile that need unique identification can use the MPEG registration_descriptor with a unique format_identifier value that is obtained from the Registration Authority.

2.6.95 HEVC video descriptor

For an HEVC video stream, the HEVC video descriptor (see Table 2-111) provides basic information for identifying coding parameters, such as profile and level parameters of that HEVC video stream. For an HEVC temporal video sub-bitstream or an HEVC temporal video subset, the HEVC video descriptor provides information such as the associated HEVC highest temporal sub-layer representation contained in the elementary stream to which it applies. This descriptor can also be used to indicate presence of WCG and HDR video components in the associated PID as well as additional parameters to assist decoders with HDR capability to render intended video data on HDR capable display devices. In addition, this can assist non-HDR capable decoders to use the information appropriately.

NOTE 1 – In case that the video characteristics change over time, care should be taken that the descriptor is updated accordingly.

This descriptor, when present, shall only be used for elementary streams with a stream_type value of 0x24 or 0x25. When the program element for which this descriptor is used is part of an HEVC layered video stream, i.e., the program contains at least one other program element with a stream_type value in the range of 0x28-0x2B, the semantics of HEVC_still_present_flag, HEVC_24hr_picture_present_flag and sub_pic_hrd_params_not_present_flag shall apply to the whole HEVC layered video stream, i.e., also to all program elements with a stream_type value in the range of 0x28-0x2B.

NOTE 2 – For elementary streams with a stream_type value in the range of 0x28-0x2B, the applicable value of level_idc can be ambiguous and depend on the output layer set, i.e., the combination with other elementary streams. This information is signalled by the HEVC operation point descriptor.

Table 2-111 – HEVC video descriptor

Syntax	No. of bits	Mnemonic
HEVC_video_descriptor() {		
descriptor_tag	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
profile_space	**2**	**uimsbf**
tier_flag	**1**	**bslbf**
profile_idc	**5**	**uimsbf**
profile_compatibility_indication	**32**	**bslbf**
progressive_source_flag	**1**	**bslbf**
interlaced_source_flag	**1**	**bslbf**
non_packed_constraint_flag	**1**	**bslbf**
frame_only_constraint_flag	**1**	**bslbf**
copied_44bits	**44**	**bslbf**
level_idc	**8**	**uimsbf**
temporal_layer_subset_flag	**1**	**bslbf**
HEVC_still_present_flag	**1**	**bslbf**
HEVC_24hr_picture_present_flag	**1**	**bslbf**
sub_pic_hrd_params_not_present_flag	**1**	**bslbf**
reserved	**2**	**bslbf**
HDR_WCG_idc	**2**	**bslbf**
if (temporal_layer_subset_flag == '1') {		
temporal_id_min	**3**	**uimsbf**
reserved	**5**	**bslbf**
temporal_id_max	**3**	**uimsbf**
reserved	**5**	**bslbf**
}		
}		

2.6.96 Semantic definition of fields in HEVC video descriptor

profile_space, tier_flag, profile_idc, profile_compatibility_indication, **progressive_source_flag, interlaced_source_flag, non_packed_constraint_flag, frame_only_constraint_flag, level_idc** – When the HEVC video descriptor applies to an HEVC video stream or to an HEVC complete temporal representation, these fields shall be coded according to the semantics defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 for *general_profile_space*, *general_tier_flag*, *general_profile_idc*, *general_profile_compatibility_flag[i]*, *general_progressive_source_flag*, *general_interlaced_source_flag*, *general_non_packed_constraint_flag*, *general_frame_only_constraint_flag*, *general_level_idc*, respectively, for the corresponding HEVC video stream or HEVC complete temporal representation, and the entire HEVC video stream or HEVC complete temporal representation to which the HEVC video descriptor is associated shall conform to the information signalled by these fields.

When the HEVC video descriptor applies to an HEVC temporal video sub-bitstream or HEVC temporal video subset of which the corresponding HEVC highest temporal sub-layer representation is not an HEVC complete temporal representation, these fields shall be coded according to the semantics defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 for *sub_layer_profile_space*, *sub_layer_tier_flag*, *sub_layer_profile_idc*, *sub_layer_profile_compatibility_flag[i]*, *sub_layer_progressive_source_flag*, *sub_layer_interlaced_source_flag*, *sub_layer_non_packed_constraint_flag*, *sub_layer_frame_only_constraint_flag*, *sub_layer_level_idc*, respectively, for the corresponding HEVC highest temporal sub-layer representation, and the entire HEVC highest temporal sub-layer representation to which the HEVC video descriptor is associated shall conform to the information signalled by these fields.

NOTE 1 – In one or more sequences in the HEVC video stream the level may be lower than the level signalled in the HEVC video descriptor, while also a profile may occur that is a subset of the profile signalled in the HEVC video descriptor. However, in the entire HEVC video stream, only subsets of the entire bitstream syntax shall be used that are included in the profile signalled in the HEVC video descriptor, if present. If the sequence parameter sets in an HEVC video stream signal different profiles, and no additional constraints are signalled, then the stream may need examination to determine which profile, if any, the entire stream conforms to. If an HEVC video descriptor is to be associated with an HEVC video stream that does not conform to a single profile, then the HEVC video stream should be partitioned into two or more sub-streams, so that HEVC video descriptors can signal a single profile for each such sub-stream.

copied_44bits – When the HEVC video descriptor applies to an HEVC video stream or to an HEVC complete temporal representation, this bit field shall be coded according to the semantics of the syntax elements defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 for the 44 bits found in the profile_tier_level() syntax element between *general_frame_only_constraint_flag* and *general_level_idc* for the corresponding HEVC video stream or HEVC complete temporal representation, and the entire HEVC video stream or HEVC complete temporal representation to which the HEVC video descriptor is associated shall conform to the information signalled by these fields.

When the HEVC video descriptor applies to an HEVC temporal video sub-bitstream or HEVC temporal video subset of which the corresponding HEVC highest temporal sub-layer representation is not an HEVC complete temporal representation, this bit field shall be coded according to the semantics of the syntax elements defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 for the 44 bits found in the profile_tier_level() syntax element between *sub_layer_frame_only_constraint_flag* and *sub_layer_level_idc* for the corresponding HEVC highest temporal sub-layer representation, and the entire HEVC highest temporal sub-layer representation to which the HEVC video descriptor is associated shall conform to the information signalled by these fields.

temporal_layer_subset_flag – This 1-bit flag, when set to '1', indicates that the syntax elements describing a subset of temporal layers are included in this descriptor. This field shall be set to 1 for HEVC temporal video subsets and for HEVC temporal video sub-bitstreams. When set to '0', the syntax elements temporal_id_min and temporal_id_max are not included in this descriptor.

HEVC_still_present_flag – This 1-bit field, when set to '1', indicates that the HEVC video stream or the HEVC highest temporal sub-layer representation may include HEVC still pictures. When the HEVC_still_present_flag is set to '0', the associated HEVC video stream shall not contain HEVC still pictures.

When the program element to which this descriptor applies is part of an HEVC layered video stream and the HEVC_still_present_flag is set to '0', the whole HEVC layered video stream shall not contain HEVC still pictures.

NOTE 2 – According to Rec. ITU-T H.265 | ISO/IEC 23008-2, IDR pictures are always associated with a TemporalId value equal to 0. Consequently, if the HEVC video descriptor applies to an HEVC temporal video subset, HEVC still pictures can only be present in the associated HEVC temporal video sub-bitstream.

HEVC_24_hour_picture_present_flag – This 1-bit flag, when set to '1', indicates that the associated HEVC video stream or the HEVC highest temporal sub-layer representation may contain HEVC 24-hour pictures. For the definition of an HEVC 24-hour picture, see 2.1.42. When the HEVC_24_hour_picture_present_flag is set to '0', the associated HEVC video stream shall not contain any HEVC 24-hour pictures.

When the program element to which this descriptor applies is part of an HEVC layered video stream and HEVC_24_hour_picture_present_flag is set to '0', the whole HEVC layered video stream shall not contain any HEVC 24-hour pictures.

sub_pic_hrd_params_not_present_flag – This 1-bit field, when set to '0', indicates that the VUI in the HEVC video stream shall have the syntax element sub_pic_hrd_params_present_flag set to '1'. When the sub_pic_hrd_params_not_present_flag is equal to '1', the associated HEVC video stream may not contain sub_pic_hrd_params_present_flag in the VUI or the sub_pic_hrd_params_present_flag may be set to '0'.

When the program element to which this descriptor applies is part of an HEVC layered video stream and sub_pic_hrd_params_not_present_flag is set to '0', the following apply:

- The HEVC timing and HRD descriptor shall be present in the program map table associated with the program.

  NOTE 3 – If sub_picture_hrd_params_not_present equals '0', HRD parameters can be expected to be present, though the hrd_management_valid_flag is not mandated to be set to '1' in this case.

- The HRD parameter structures that are applicable for all program elements with stream_type value of 0x24, 0x25, or in the range of 0x28-0x2B, inclusively, shall be present in the HEVC video stream and the value of sub_pic_hrd_params_present_flag in those HRD parameter structures shall be set to '1'.

HDR_WCG_idc – The value of this syntax element indicates the presence or absence of high dynamic range (HDR) and/or wide color gamut (WCG) video components in the associated PID according to Table 2-112. HDR is defined to be video that has high dynamic range if the video stream EOTF is higher than the Rec. ITU-R BT.1886 reference EOTF. This field also shall not be set to 2 unless bit_depth_luma_minus8 as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 in the associated video is greater than or equal to 2. WCG is defined to be video that is coded using colour primaries with a colour gamut not contained within Rec. ITU-R BT.709. This field also shall not be set to 1 or 2 unless bit_depth_chroma_minus8 as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 in the associated video is greater than or equal to 2.

Table 2-112 – Semantics of HDR_WGC_idc

HDR_WCG_idc	Description
0	SDR, i.e., video is based on the Rec. ITU-R BT.1886 reference EOTF with a color gamut that is contained within Rec. ITU-R BT.709 with a Rec. ITU-R BT.709 container (see Note 1)
1	WCG only, i.e., video color gamut in a Rec ITU-R BT.2020 container that exceeds Rec. ITU-R BT.709 (see Note 2)
2	Both HDR and WCG are to be indicated in the stream (see Note 3)
3	No indication made regarding HDR/WCG or SDR characteristics of the stream
NOTE 1 – An example where it would be desirable to set HDR_WCG_idc to 0 would be when the colour_description_present_flag, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2, is set to '0', with colour_primaries and transfer_characteristics not present in the video stream. NOTE 2 – An example where it would be desirable to set HDR_WCG_idc to 1 would be when colour_primaries as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 is equal to 9 to indicate Rec. ITU-R BT.2020. NOTE 3 – An example where it would be desirable to set HDR_WCG_idc to 2 would be when transfer_characteristics as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 is equal to 16 to indicate BT.2100 PQ EOTF or equal to 18 to indicate BT.2100 HLG EOTF, and when colour_primaries as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 is equal to 9 to indicate Rec. ITU-R BT.2020.	

temporal_id_min – This 3-bit field indicates the minimum value of the *TemporalId*, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2, of all HEVC access units in the associated elementary stream.

temporal_id_max – This 3-bit field indicates the maximum value of the *TemporalId*, as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2, of all HEVC access units in the associated elementary stream.

2.6.97 HEVC timing and HRD descriptor

For an HEVC video stream, an HEVC temporal video sub-bitstream or an HEVC temporal video subset, the HEVC timing and HRD descriptor (see Table 2-113) provides timing and HRD parameters, as defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2, for the associated HEVC video stream or the HEVC highest temporal sub-layer representation thereof, respectively.

Table 2-113 – HEVC timing and HRD descriptor

Syntax	No. Of bits	Mnemonic
HEVC_timing_and_HRD_descriptor() {		
hrd_management_valid_flag	**1**	**bslbf**
target_schedule_idx_not_present_flag	**1**	**bslbf**
target_schedule_idx	**5**	**uimsbf**
picture_and_timing_info_present_flag	**1**	**bslbf**
if (picture_and_timing_info_present_flag == '1') {		
90kHz_flag	**1**	**bslbf**
reserved	**7**	**bslbf**
if (90kHz_flag = = '0') {		
N	**32**	**uimsbf**
K	**32**	**uimsbf**
}		
num_units_in_tick	**32**	**uimsbf**
}		
}		

2.6.98 Semantic definition of fields in HEVC timing and HRD descriptor

hrd_management_valid_flag – This 1-bit flag is only defined for use in transport streams. When the HEVC timing and HRD descriptor is associated with an HEVC video stream or with an HEVC highest temporal sub-layer representation carried in a transport stream, then the following rules apply.

When the value of hrd_management_valid_flag is equal to '1', Buffering Period SEI and Picture Timing SEI messages, as defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2, shall be present in the associated HEVC video stream or HEVC highest temporal sub-layer representation. For HEVC layered video streams, each HEVC operation point signalled in the HEVC operation point descriptor shall have applicable Buffering Period SEI and Picture Timing SEI messages. All Buffering Period SEI messages shall carry coded nal_initial_cpb_removal_delay and nal_initial_cpb_removal_offset values and may additionally carry nal_initial_alt_removal_delay and nal_initial_alt_cpb_removal_offset values for the NAL HRD. If the hrd_management_valid_flag is set to '1', then the transfer of each byte from MB_n to EB_n in the T-STD as defined in 2.17.2 or the transfer from $MB_{n,k}$ to EB_n in the T-STD as defined in 2.17.3 or the transfer of each byte from $MB_{l,k}$ to EB_l in the T-STD as defined in 2.17.4 shall be according to the delivery schedule for that byte into the CPB in the NAL HRD, as determined from the coded nal_initial_cpb_removal_delay and nal_initial_cpb_removal_offset or from the coded nal_initial_alt_cpb_removal_delay and nal_initial_alt_cpb_removal_offset values for SchedSelIdx equal to target_schedule_idx, as specified in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2. When the hrd_management_valid_flag is set to '0', the leak method shall be used for the transfer from MB_n to EB_n in the T-STD as defined in 2.17.2 or the transfer from $MB_{n,k}$ to EB_n in the T-STD as defined in 2.17.3 or the transfer from $MB_{l,k}$ to EB_l in the T-STD as defined in 2.17.4.

target_schedule_idx_not_present_flag – This 1-bit flag when set to '0' indicates that the following 5 bits represent the value target_schedule_idx as specified below. When set to '1', the following 5 bits are unspecified. When hrd_management_valid_flag is equal to 0, then target_schedule_idx_not_present_flag shall be set to '1'.

target_schedule_idx – When target_schedule_idx_not_present_flag is equal to '0', this 5-bit field indicates the index of the delivery schedule which is assigned for SchedSelIdx. When the value of target_schedule_idx_not_present_flag is equal to '1' and the value of hrd_management_valid_flag is equal to '1', the value of target_schedule_idx is inferred to be equal to '0'.

picture_and_timing_info_present_flag – This 1-bit flag when set to '1' indicates that the *90kHz_flag* and parameters for accurate mapping to a 90-kHz system clock are included in this descriptor.

90kHz_flag – This 1-bit flag when set to '1' indicates that the frequency of the HEVC time base is 90 kHz.

N, K – For an HEVC video stream or HEVC highest temporal sub-layer representation, the frequency of the HEVC time base is defined by the syntax element *vui_time_scale* in the VUI parameters, as defined in Annex E of Rec. ITU-T H.265 | ISO/IEC 23008-2. The relationship between the HEVC *time_scale* and the STC shall be defined by the parameters N and K in this descriptor as follows.

$$time\_scale = (N \times system\_clock\_frequency) / K$$

If the *90kHz_flag* is set to '1', then N equals 1 and K equals 300. If the *90kHz_flag* is set to '0', then the values of N and K are provided by the coded values of the N and K fields.

NOTE – This allows mapping of time expressed in units of *time_scale* to 90 kHz units, as needed for the calculation of PTS and DTS timestamps, for example in decoders for HEVC access units for which no PTS or DTS is encoded in the PES header.

num_units_in_tick – This 32-bit field is coded exactly in the same way as the *vui_num_units_in_tick* field in VUI parameters in Annex E of Rec. ITU-T H.265 | ISO/IEC 23008-2. The information provided by this field shall apply to the entire HEVC video stream or HEVC highest temporal sub-layer representation to which the HEVC timing and HRD descriptor is associated.

2.6.99 AF extensions descriptor

The AF extensions descriptor (see Table 2-114) is used to signal that adaptation field descriptors could be present in the adaptation header of the component, as defined in 2.4.3.5.

NOTE – There may be AF descriptors in an adaptation field of a TS packet even though this descriptor is not set for the component.

Table 2-114 – Adaptation field extension descriptor

Syntax	No. of bits	Mnemonic
af_extensions_descriptor() { }		

2.6.100 HEVC operation point descriptor

The HEVC operation point descriptor (see Table 2-115) provides a method to indicate profile and level for one or more HEVC operation points. When present, the HEVC operation point descriptor shall be included in the group of data elements which immediately follow the program_info_length field in the program_map section.

NOTE – For some applications, the TS may not contain all operation points described in the HEVC operation point descriptor, or the HEVC operation point descriptor may not describe all operation points available in the TS. However, as far as matching elementary streams are found in the TS, the information provided in the descriptor should describe the operation points correctly.

Table 2-115 – HEVC operation point descriptor

Syntax	No. of bits	Mnemonic
HEVC_operation_point_descriptor( ) {		
reserved	**2**	**bslbf**
num_ptl	**6**	**uimsbf**
for (i = 0; i < num_ptl; i++, i++) {		
profile_tier_level_info[i]	**96**	**bslbf**
}		
operation_points_count	**8**	**uimsbf**
for (i = 0; i < operation_points_count; i++) {		
target_ols[i]	**8**	**uimsbf**
ES_count[i]	**8**	**uimsbf**
for (j = 0; j < ES_count[i]; j++) {		
reserved	**1**	**bslbf**
prepend_dependencies[i][j]	**1**	**bslbf**
ES_reference[i][j]	**6**	**uimsbf**
}		
reserved	**2**	**bslbf**
numEsInOp[i]	**6**	**uimsbf**
for (k = 0; k < NumESinOP[i]; k++) {		
necessary_layer_flag[i][k]	**1**	**bslbf**
output_layer_flag[i][k]	**1**	**bslbf**
ptl_ref_idx[i][k]	**6**	**uimsbf**
}		
reserved	**1**	**bslbf**
avg_bit_rate_info_flag[i]	**1**	**bslbf**
max_bit_rate_info_flag[i]	**1**	**bslbf**
constant_frame_rate_info_idc[i]	**2**	**uimsbf**
applicable_temporal_id[i]	**3**	**uimsbf**
if (constant_frame_rate_info_idc[i] > 0) {		
reserved	**4**	**bslbf**
frame_rate_indicator[i]	**12**	**uimsbf**
}		
if (avg_bit_rate_info_flag[i] == '1') {		
avg_bit_rate[i]	**24**	**uimsbf**
}		
if (max_bit_rate_info_flag[i] == '1') {		
max_bit_rate[i]	**24**	**uimsbf**
}		
}		
}		

2.6.101 Semantic definition of fields in HEVC operation point descriptor

num_ptl – This 6-bit field specifies the number of profile, tier and level structures signalled in this descriptor.

profile_tier_level_info[i] – This 96-bit field shall be coded according to the syntax structure of profile_tier_level defined in 7.3.3 of Rec. ITU-T H.265 | ISO/IEC 23008-2 with the value of profilePresentFlag set equal to '1' and maxNumSubLayersMinus1 set equal to 6.

If multiple HEVC operation point descriptors are found for the same program, all profile_tier_level_info[x] elements of all HEVC operation point descriptors for this program are aggregated in their order of occurrence into a common array, which is referenced in this specification as profile_tier_level_array[]. If there is only a single HEVC operation point descriptor, profile_tier_level_array[] contains the elements profile_tier_level_info[x] in the order as found in that single descriptor.

operation_points_count – This 8-bit field indicates the number of HEVC operation points described by the list included in the following group of data elements.

target_ols[i] – An 8-bit field that specifies the index into the list of output layer sets in the VPS, associated with the i-th HEVC operation point defined in this descriptor.

ES_count[i] – This 8-bit field indicates the number of ES_reference values included in the following group of data elements. The aggregation of elementary streams, according to the ordered list indicated in the following group of data elements, forms an HEVC operation point. The value 0xff is reserved.

Let OperationPointESList[i] be the list of elementary streams that are part of the i-th HEVC operation point.

prepend_dependencies[i][j] – This flag if set to '1' specifies that the elementary stream indicated by ES_reference[i][j], when not present yet in OperationPointESList[i], shall be added into OperationPointESList[i] and the elementary stream indicated by the syntax element hierarchy_embedded_layer_index in the hierarchy descriptor, or all of the elementary streams indicated by the syntax element hierarchy_ext_embedded_layer_index in the HEVC hierarchy extension descriptor, with the hierarchy layer index value specified by the following syntax element ES_reference[i][j], when not present yet in OperationPointESList[i], shall be added into OperationPointLayerList[i] immediately before the elementary stream signalled by the ES_reference[i][j] in ascending order of the value of their associated hierarchy_embedded_layer_index or hierarchy_ext_embedded_layer_index. When the value of prepend_dependencies[i][j] is equal to '0', only the elementary stream indicated by ES_reference[i][j], when not present yet in OperationPointESList[i], shall be added into OperationPointESList[i]. The elementary stream indicated by ES_reference[i][m] shall be placed earlier (i.e., with a lower index) into OperationPointESList[i] than the elementary stream indicated with ES_reference[i][n] when m is less than n. The order of elementary stream in the OperationPointESList[i] shall be in ascending order of their hierarchy_layer_index values.

ES_reference[i][j] – This 6-bit field indicates the hierarchy layer index value present in the hierarchy descriptor or HEVC hierarchy extension descriptor which identifies an elementary stream. The value of ES_reference[i][m] and ES_reference[i][n] for m not equal to n shall not be the same.

numEsInOp[i] – This 6-bit field indicates the number of elementary streams in OperationPointESList[i] after all the ESs that are part of the i-th HEVC operation point have been included into OperationPointESList[i] (i.e., after parsing prepend_dependencies[i][ES_count[i] – 1]).

necessary_layer_flag[i][k] – This flag when set to '1' indicates that the k-th elementary stream in OperationPointESList[i] is a necessary layer, as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, of the i-th operation point. This flag equal to '0' indicates that the k-th elementary stream in OperationPointESList[i] is not a necessary layer, as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, of the i-th operation point.

output_layer_flag[i][k] – This flag when set to '1' indicates that the k-th elementary stream in OperationPointESList[i] is an output layer. Otherwise, when set to '0', it indicates that the k-th elementary stream in OperationPointESList[i] is not an output layer. When the value of necessary_layer_flag[i][k] is equal to '0', the value of output_layer_flag[i][k] shall be ignored.

ptl_ref_idx[i][k] – A 6-bit field that indicates the index x to the profile_tier_level_info[x] element of the profile_tier_level_array which applies to the k-th elementary stream in OperationPointESList[i]. When the value of necessary_layer_flag[i][k] is equal to '0', the value of ptl_ref_idx[i][k] shall be ignored.

avg_bit_rate_info_flag[i] – This flag indicates whether the syntax element avg_bit_rate[i] is present in this descriptor.

max_bit_rate_info_flag[i] – This flag indicates whether the syntax element max_bit_rate[i] is present in this descriptor.

constant_frame_rate_info_idc[i] – This 2-bit field, in combination with the syntax element frame_rate_indicator as specified below, indicates how the frame rate for the associated operation point j is determined. The value of 0 indicates that the frame rate is not specified for the i-th HEVC operation point and that the syntax element frame_rate_indicator is not present in this descriptor for the i-th HEVC operation point.

applicable_temporal_id[i] – This 3-bit field indicates the highest value of TemporalId of the VCL NAL units in the re-assembled HEVC video stream for operation point i.

frame_rate_indicator[i] – If constant_frame_rate_info_idc[i] is equal to 1, this 12-bit field indicates a constant number of ticks, as specified in the HEVC timing and HRD descriptor, for the distance in time between two pictures at the i-th HEVC operation point. If constant_frame_rate_info_idc[i] equals 2, this 12-bit field indicates the frame rate for the i-th operation point measured in frames per second. If constant_frame_rate_info_idc[i] equals 3, this 12-bit field indicates the frame rate for the i-th HEVC operation point measured in frames per 1.001 seconds.

avg_bit_rate[i] – This 24-bit field indicates the average bit rate, in 1000 bits per second, of the HEVC layered video stream corresponding to the i-th HEVC operation point.

max_bit_rate[i] – This 24-bit field indicates the maximum bit rate, in 1000 bits per second, of the HEVC layered video stream corresponding to the i-th HEVC operation point.

2.6.102 HEVC hierarchy extension descriptor

The HEVC hierarchy extension descriptor provides information to identify the program elements containing components of layered HEVC streams (see Table 2-116). When present, this descriptor shall only be used for elementary streams with the stream_type value 0x28, 0x29, 0x2A or 0x2B.

Table 2-116 – HEVC hierarchy extension descriptor

Syntax	No. of bits	Mnemonic
HEVC_hierarchy_extension_descriptor( ) {		
extension_dimension_bits	**16**	**bslbf**
hierarchy_layer_index	**6**	**uimsbf**
temporal_id	**3**	**uimsbf**
nuh_layer_id	**6**	**uimsbf**
tref_present_flag	**1**	**bslbf**
Reserved	**2**	**bslbf**
num_embedded_layers	**6**	**uimsbf**
Reserved	**2**	**bslbf**
hierarchy_channel	**6**	**uimsbf**
for (i = 0 ; i < num_embedded_layers ; i++) {		
Reserved	**2**	**bslbf**
hierarchy_ext_embedded_layer_index[i]	**6**	**uimsbf**
}		
}		

2.6.103 Semantic definition of fields in HEVC hierarchy extension descriptor

When the HEVC hierarchy extension descriptor is present, it is used to specify the dependency of the associated elementary stream to other elementary streams in the same program.

extension_dimension_bits – A 16-bit field indicating the possible enhancement of the associated program element from the base layer resulting from the program element of the layer with nuh_layer_id equal to '0'.

The allocation of the bits to enhancement dimensions is given in Table 2-117.

Table 2-117 – Semantics of extension dimension bits

Index to bits	Description
0	Multi-view enhancement
1	Spatial scalability, including SNR quality or fidelity enhancement
2	Depth enhancement
3	Temporal enhancement
4	Auxiliary enhancement
5 .. 15	Reserved

The i-th bit equal to '1' indicates that the corresponding enhancement dimension is present. When the elementary stream contains auxiliary pictures as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, the value of the 4th bit of extension_dimension_bits shall be set equal to '1', otherwise, it shall be set equal to '0'. When the elementary stream contains auxiliary pictures that are depth pictures, as defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, the value of both the 2nd and the 4th bits of extension_dimension_bits shall be set equal to '1'.

hierarchy_layer_index – A 6-bit field that defines a unique index of the associated program elements in a table of coding layer hierarchies. Indices shall be unique within a single program definition. For video sub-bitstreams of HEVC video streams conforming to one or more profiles defined in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, this is the program element index, which is assigned in a way that the bitstream order will be correct if the associated dependency layers of the video sub-bitstreams of the same HEVC access unit are re-assembled in increasing order of hierarchy_layer_index.

temporal_id – A 3-bit field that specifies the highest TemporalId of the NAL units in the elementary stream associated with this HEVC hierarchy extension descriptor.

nuh_layer_id – A 6-bit field that specifies the highest nuh_layer_id of the NAL units in the elementary stream associated with this HEVC hierarchy extension descriptor.

tref_present_flag – A 1-bit flag, which when set to '0' indicates that the TREF field may be present in the PES packet headers in the associated elementary stream. The value of '1' for this flag is reserved.

num_embedded_layers – A 6-bit field that specifies the number of direct dependent program elements that need to be accessed and be present in decoding order before decoding of the elementary stream associated with this HEVC hierarchy extension descriptor.

hierarchy_channel – A 6-bit field that indicates the intended channel number for the associated program element in an ordered set of transmission channels. The most robust transmission channel is defined by the lowest value of this field with respect to the overall transmission hierarchy definition.

NOTE – A given hierarchy_channel may at the same time be assigned to several program elements.

hierarchy_ext_embedded_layer_index[i] – A 6-bit field that defines the hierarchy_layer_index of the program element that needs to be accessed and be present in decoding order before decoding of the elementary stream associated with this HEVC hierarchy extension descriptor.

2.6.104 Green extension descriptor

The syntax of the green extension descriptor containing static metadata is shown in Table 2-118.

Table 2-118 – Green extension descriptor

Syntax	No. bits	Mnemonic
Green_extension_descriptor() {		
num_constant_backlight_voltage_time_intervals	**2**	**uimsbf**
reserved	**6**	**bslbf**
for (i=0; i < num_constant_backlight_voltage_time_intervals; i++) {		
constant_backlight_voltage_time_interval[i]	**16**	**uimsbf**
}		
num_max_variations	**2**	**uimsbf**
reserved	**6**	**bslbf**
for (j=0; j < num_max_variations; j++) {		
max_variation[j]	**16**	**uimsbf**
}		
}		

2.6.105 Semantics for green extension descriptor

Semantics for all the syntax elements above are specified in 6.4 of ISO/IEC 23001-11.

2.6.106 MPEG-H 3D audio descriptor

The MPEG-H 3D audio descriptor (see Table 2-119) provides information on basic coding information in the associated ISO/IEC 23008-3 stream. This descriptor shall be present in the associated PMT for MPEG-H 3D audio content with stream_type equal to 0x2D.

Table 2-119 – MPEG-H 3D audio descriptor

Syntax	No of bits	Mnemonic
MPEG-H_3dAudio_descriptor() {		
mpegh3daProfileLevelIndication	**8**	**uimsbf**
interactivityEnabled	**1**	**bslbf**
reserved	**9**	**bslbf**
referenceChannelLayout	**6**	**uimsbf**
for (i=0; i<N; i++) {		
reserved	**8**	**bslbf**
}		
}		

2.6.107 Semantics for MPEG-H 3D audio descriptor

mpegh3daProfileLevelIndication – The audio profile and level of the associated ISO/IEC 23008-3 audio stream, encoded as specified for the mpegh3daProfileLevelIndication field in 5.3.2 in ISO/IEC 23008-3.

referenceChannelLayout – Reference channel configuration value as defined as "ChannelConfiguration" in ISO/IEC 23001-8 ("Codec Independent Code Points").

interactivityEnabled – If set to 1, this flag indicates that the 3D audio stream contains elements with associated metadata which enables user interactivity. If this flag is set to 0, no user interactivity of any kind is available. This flag may be used to determine the need for initializing the user interactivity interface in the Systems decoder.

2.6.108 MPEG-H 3D audio config descriptor

The MPEG-H 3D audio config descriptor (see Table 2-120) provides information on the complete configuration data of one ISO/IEC 23008-3 stream.

Table 2-120 – MPEG-H 3D audio config descriptor

Syntax	No of bits	Mnemonic
MPEG-H_3dAudio_config_descriptor() {		
mpegh3daConfig()		
}		

2.6.109 Semantics for MPEG-H 3D audio config descriptor

mpegh3daConfig() – The mpegh3daConfig() of the associated ISO/IEC 23008-3 audio stream, as specified in 5.2.2.1 in ISO/IEC 23008-3.

2.6.110 MPEG-H 3D audio scene descriptor

The MPEG-H 3D audio scene descriptor (see Table 2-121) provides information on user selectable and/or modifiable audio objects in an ISO/IEC 23008-3 stream.

Table 2-121 – MPEG-H 3D audio scene descriptor

Syntax	No of bits	Mnemonic
MPEG-H_3dAudio_scene_descriptor() {		
groupDefinitionPresent	**1**	**bslbf**
switchGroupDefinitionPresent	**1**	**bslbf**
groupPresetDefinitionPresent	**1**	**bslbf**
reserved	**5**	**bslbf**
3dAudioSceneInfoID	**8**	**bslbf**
if (groupDefinitionPresent) {		
reserved	**1**	**bslbf**
numGroups	**7**	**uimsbf**
for (i=0; i < numGroups; i++) {		
reserved	**1**	**bslbf**
mae_groupID	**7**	**uimsbf**
reserved	**3**	**bslbf**
mae_allowOnOff	**1**	**bslbf**
mae_defaultOnOff	**1**	**bslbf**
mae_allowPositionInteractivity	**1**	**bslbf**
mae_allowGainInteractivity	**1**	**bslbf**
mae_hasContentLanguage	**1**	**bslbf**
reserved	**4**	**bslbf**
mae_contentKind	**4**	**uimsbf**
if (mae_allowPositionInteractivity) {		

Syntax	No of bits	Mnemonic
reserved	**1**	**bslbf**
mae_interactivityMinAzOffset	**7**	**uimsbf**
reserved	**1**	**bslbf**
mae_interactivityMaxAzOffset	**7**	**uimsbf**
reserved	**3**	**bslbf**
mae_interactivityMinElOffset	**5**	**uimsbf**
reserved	**3**	**bslbf**
mae_interactivityMaxElOffset	**5**	**uimsbf**
mae_interactivityMinDistOffset	**4**	**uimsbf**
mae_interactivityMaxDistOffset	**4**	**uimsbf**
}		
if (mae_allowGainInteractivity) {		
reserved	**2**	**bslbf**
mae_interactivityMinGain	**6**	**uimsbf**
reserved	**3**	**bslbf**
mae_interactivityMaxGain	**5**	**uimsbf**
}		
if (mae_hasContentLanguage) {		
mae_contentLanguage	**24**	**uimsbf**
}		
}		
}		
if (switchGroupDefinitionPresent) {		
reserved	**3**	**bslbf**
numSwitchGroups	**5**	**uimsbf**
for (i=0; i < numSwitchGroups; i++) {		
reserved	**1**	**bslbf**
mae_switchGroupID	**5**	**uimsbf**
mae_switchGroupAllowOnOff	**1**	**bslbf**
mae_switchGroupDefaultOnOff	**1**	**bslbf**
reserved	**3**	**bslbf**
mae_bsSwitchGroupNumMembers	**5**	**uimsbf**
for (i = 0; i < mae_bsSwitchGroupNumMembers + 1; i++) {		
reserved	**1**	**bslbf**
mae_switchGroupMemberID	**7**	**uimsbf**
}		
reserved	**1**	**bslbf**
mae_switchGroupDefaultGroupID	**7**	**uimsbf**
}		
}		
if (presetGroupDefinitionPresent) {		
reserved	**3**	**bslbf**
mae_numGroupPresets	**5**	**uimsbf**
for (i = 0; i < mae_numGroupPresets; i++) {		
reserved	**3**	**bslbf**
mae_groupPresetID	**5**	**uimsbf**
reserved	**3**	**bslbf**
mae_groupPresetKind	**5**	**uimsbf**
reserved	**4**	**bslbf**
mae_numGroupPresetConditions	**4**	**uimsbf**

Syntax	No of bits	Mnemonic
for (j = 0; j < mae_numGroupPresetConditions+1; j++) {		
mae_groupPresetGroupID	**7**	**uimsbf**
mae_groupPresetConditionOnOff	**1**	**bslbf**
if (mae_groupPresetConditionOnOff) {		
reserved	**4**	**bslbf**
mae_groupPresetDisableGainInteractivity	**1**	**bslbf**
mae_groupPresetGainFlag	**1**	**bslbf**
mae_groupPresetDisablePositionInteractivity	**1**	**bslbf**
mae_groupPresetPositionFlag	**1**	**bslbf**
if (mae_groupPresetGainFlag) {		
mae_groupPresetGain	**8**	**uimsbf**
}		
if(mae_groupPresetPositionFlag){		
mae_groupPresetAzOffset	**8**	**uimsbf**
reserved	**2**	**bslbf**
mae_groupPresetElOffset	**6**	**uimsbf**
reserved	**4**	**bslbf**
mae_groupPresetDistFactor	**4**	**uimsbf**
}		
}		
}		
}		
}		
for (i=0; i<N; i++) {		
reserved	**8**	**bslbf**
}		
}		

2.6.111 Semantic definition of fields in MPEG-H 3D audio scene descriptor

groupDefinitionPresent – A one-bit flag signalling the presence of interactivity information of one group in this descriptor.

groupContentDataPresent – A one-bit flag signalling the presence of content information of one group in this descriptor.

switchGroupDefinitionPresent – A one-bit flag signalling the presence of switch group information in this descriptor.

presetGroupDefinitionPresent – A one-bit flag signalling the presence of preset group information in this descriptor.

3dAudioSceneInfoID – See 15.3 of ISO/IEC 23008-3.

numGroups – This field signals the number of groups in the audio scene description. This field can take values between 1 and 127, and shall be less or equal to the value of mae_numGroups present in the associated ISO/IEC 23008-3 stream.

mae_groupID – See 15.3 of ISO/IEC 23008-3.

mae_allowOnOff – See 15.3 of ISO/IEC 23008-3.

mae_defaultOnOff – See 15.3 of ISO/IEC 23008-3.

mae_allowPositionInteractivity – See 15.3 of ISO/IEC 23008-3.

mae_allowGainInteractivity – See 15.3 of ISO/IEC 23008-3.

mae_hasContentLanguage – See 15.3 of ISO/IEC 23008-3.

mae_contentKind – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMinAzOffset – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMaxAzOffset – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMinElOffset – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMaxElOffset – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMinDistOffset – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMaxDistOffset – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMinGain – See 15.3 of ISO/IEC 23008-3.

mae_interactivityMaxGain – See 15.3 of ISO/IEC 23008-3.

mae_contentLanguage – See 15.3 of ISO/IEC 23008-3.

numSwitchGroups – This field signals the number of switch groups minus one in the overall scene. This field can take values between 0 and 31, resulting in a maximum number of 32 switch groups. It shall be less or equal to the value of mae_numSwitchGroups present in the associated ISO/IEC 23008-3 stream.

mae_switchGroupID – See ISO/IEC 23008-3.

mae_switchGroupAllowOnOff – See 15.3 of ISO/IEC 23008-3.

mae_switchGroupDefaultOnOff – See 15.3 of ISO/IEC 23008-3; if mae_switchGroupAllowOnOff is '0', then mae_switchGroupDefaultOnOff shall be set to '0'.

mae_bsSwitchGroupNumMembers – See 15.3 of ISO/IEC 23008-3.

mae_switchGroupMemberID – See 15.3 of ISO/IEC 23008-3.

mae_switchGroupDefaultGroupID – See 15.3 of ISO/IEC 23008-3.

mae_numGroupPresets – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetID – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetKind – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetNumConditions – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetGroupID – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetConditionOnOff – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetDisableGainInteractivity – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetGainFlag – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetDisablePositionInteractivity – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetPositionFlag – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetGain – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetAzOffset – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetElOffset – See 15.3 of ISO/IEC 23008-3.

mae_groupPresetDistFactor – See 15.3 of ISO/IEC 23008-3.

Data fields provided both in this descriptor and as in-band information in the ISO/IEC 23008-3 stream shall be set to the same value.

2.6.112 MPEG-H 3D audio text label descriptor

The MPEG-H 3D audio scene descriptor provides text labels for the audio objects and presets in an ISO/IEC 23008-3 stream. See Table 2-122.

Table 2-122 – MPEG-H 3D audio text label descriptor

Syntax	No of bits	Mnemonic
MPEG-H_3dAudio_text_label_descriptor() {		
3dAudioSceneInfoID	**8**	**uimsbf**

Syntax	No of bits	Mnemonic
reserved	**4**	**bslbf**
numDescLanguages	**4**	**uimsbf**
for (i=0; i< numDescLanguage; i++) {		
descriptionLanguage	**24**	**uimsbf**
reserved	**1**	**bslbf**
numGroupDescriptions	**7**	**uimsbf**
for (n = 0; n < numGroupDescriptions; n++) {		
reserved	**1**	**bslbf**
mae_descriptionGroupID;	**7**	**uimsbf**
groupDescriptionDataLength	**8**	**uimsbf**
for (c = 0; c < groupDescriptionDataLength; c++) {		
groupDescriptionData	**8**	**uimsbf**
}		
}		
reserved	**3**	**bslbf**
numSwitchGroupDescriptions	**5**	**uimsbf**
for (n = 0; n < numSwitchGroupDescriptions; n++) {		
reserved	**3**	**bslbf**
mae_descriptionSwitchGroupID;	**5**	**uimsbf**
switchGroupDescriptionDataLength	**8**	**uimsbf**
for (c = 0; c < switchGroupDescriptionDataLength; c++) {		
switchGroupDescriptionData	**8**	**uimsbf**
}		
}		
reserved	**3**	**bslbf**
numGroupPresetsDescriptions	**5**	**uimsbf**
for (n = 0; n < numGroupPresetsDescriptions; n++) {		
reserved	**3**	**bslbf**
mae_descriptionGroupPresetID	**5**	**uimsbf**
groupPresetDescriptionDataLength	**8**	**uimsbf**
for (c = 0; c < groupPresetDescriptionLength; c++) {		
groupPresetDescriptionData	**8**	**uimsbf**
}		
}		
}		
for (i=0; i<N; i++) {		
reserved	**8**	**bslbf**
}		
}		

2.6.113 Semantic definition of fields in MPEG-H 3D audio text label descriptor

3dAudioSceneInfoID – See 15.3 ISO/IEC 23008-3.

maeGroupDescriptionPresent – A one-bit flag signalling the presence of description text for groups.

maeSwitchgroupDescriptionPresent – A one-bit flag signalling the presence of description text for switch groups.

maeGroupPresetDescriptionPresent – A one-bit flag signalling the presence of description text for group presets.

numDescLanguages – The number of available languages for description text.

descriptionLanguage – Identifies the language or languages used by the description text of a metadata element group. It contains a 3-character code as specified by ISO 639-2. Both ISO 639-2/B and ISO 639-2/T may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 and inserted in order into the 24-bit field.

numGroupDescriptions – The number of available descriptions for groups.

mae_descriptionGroupID – See 15.3 of ISO/IEC 23008-3.

groupDescriptionDataLength – The length, specified in bytes, of the following group description.

groupDescriptionData – This field contains a description of a metadata element group, i.e., a string describing the content by a high-level description. The format shall follow UTF-8 according to ISO/IEC 10646.

numSwitchGroupDescriptions – The number of available descriptions for switch groups.

mae_descriptionSwitchGroupID – See 15.3 of ISO/IEC 23008-3.

switchGroupDescriptionDataLength – The length, specified in bytes, of the following switch group description.

switchGroupDescriptionData – This field contains a description of a switch group, i.e., a string describing the content by a high-level description. The format shall follow UTF-8 according to ISO/IEC 10646.

numGroupPresetsDescriptions – The number of available descriptions for group presets.

mae_descriptionGroupPresetID – See 15.3 of ISO/IEC 23008-3.

groupPresetDescriptionDataLength – The length, specified in bytes, of the following group preset description.

groupPresetDescriptionData – This field contains a description of a metadata element group, i.e., a string describing the content by a high-level description. The format shall follow UTF-8 according to ISO/IEC 10646.

Data fields provided both in this descriptor and as in-band information in the IS/IEC 23008-3 stream shall be set to the same value.

2.6.114 MPEG-H 3D audio multi-stream descriptor

The MPEG-H 3D audio multi-stream descriptor (see Table 2-123) provides information on the location of each mae_groupID in case of transmission over multiple streams.

In combination with this descriptor, an MPEG-H 3D audio scene descriptor explaining the actual representation of the overall audio scene is required to be present in the descriptor loop of the main stream.

Table 2-123 – MPEG-H 3D audio multi-stream descriptor

Syntax	No of bits	Mnemonic
MPEG-H_3dAudio_multi-stream_descriptor() {		
thisIsMainStream	**1**	**bslbf**
thisStreamID	**7**	**uimsbf**
if (thisIsMainStream) {		
reserved	**1**	**slbf**
numAuxiliaryStreams	**7**	**uimsbf**
reserved	**1**	**bslbf**
mae_numGroups	**7**	**uimsbf**
for (i=0; i< mae_numGroups; i++) {		
mae_groupID	**7**	**uimsbf**
isInMainStream	**1**	**bslbf**
if (thisIsMainStream == '0') {		
isInTS	**1**	**bslbf**
auxiliaryStreamID	**7**	**uimsbf**
}		
}		
}		
for (i=0; i<N; i++) {		
reserved	**8**	**bslbf**
}		
}		

2.6.115 Semantic definition of fields in MPEG-H 3D audio multi-stream descriptor

thisIsMainStream – If this flag is '1', the stream is a main stream, otherwise it is an auxiliary stream.

thisStreamID – This integer provides a unique ID of all available ISO/IEC 23008-3 Audio streams with MHAS transport syntax, both main and auxiliary streams (stream_type 0x2D and 0x2E).

numAuxiliaryStreams – This integer provides information on how many auxiliary stream are available.

mae_numGroups – This field signals the number of groups in the overall audio scene (complete number of groups in the main stream plus all possible additional streams). This field can take values between 1 and 127. It shall be set to the same value as the corresponding field in the associated ISO/IEC 23008-3 stream.

mae_groupID – This integer provides information on the mae_groupID (as described in ISO/IEC 23008-3, section 15) the loop instance refers to.

isInMainStream – If this flag is set to '1', the audio data related to the group (as indicated through mae_groupID) is present in the main stream, otherwise the encoded data is transmitted in an auxiliary stream.

isInTS – If this flag is set to '1', the audio data related to the group (as indicated through mae_groupID) is present in the same transport stream. If this flag is set to 0, the data must be retrieved from an external source.

auxiliaryStreamID – In case of transmission of encoded audio data as identified by groupID in an auxiliary stream, this integer identifies the used auxiliary stream.

The location of the 'external source' may be signalled using the TEMI location descriptor and/or the TEMI BaseURL_descriptor as defined in section U.3.

When the TEMI descriptor(s) are conveyed in a TEMI_AU as a separate elementary stream on a separated PID, an MPEG-H_3dAudio_extStreamID_descriptor() shall be present in the associated descriptor loop of the TEMI elementary stream which provides an ID in the field "auxiliaryStreamID" that is matching the "auxiliaryStreamID" provided in the MPEG-H_3dAudio_multi-stream_descriptor() of the main stream.

2.6.116 MPEG-H 3D audio DRC and Loudness descriptor

The MPEG-H 3D audio dynamic range control (DRC) and Loudness descriptor (see Table 2-124) provides information on DRC and Loudness information contained in an ISO/IEC 23008-3 stream.

Table 2-124 – MPEG-H 3D audio DRC and Loudness descriptor()

Syntax	No of bits	Mnemonic
MPEG-H_3dAudio_drc_loudness_descriptor () {		
reserved	**7**	**bslbf**
mpegh3daDrcAndLoudnessInfoPresent	**1**	**bslbf**
if (mpegh3daDrcAndLoudnessInfoPresent) {		
reserved	**2**	**bslbf**
drcInstructionsUniDrcCount	**6**	**uimsbf**
reserved	**2**	**bslbf**
loudnessInfoCount	**6**	**uimsbf**
reserved	**3**	**bslbf**
downmixIdCount	**5**	**uimsbf**
for (i=0; i<drcInstructionsUniDrcCount; i++) {		
reserved	**6**	**bslbf**
drcInstructionsType	**2**	**uimsbf**
if (drcInstructionsType == 2) {		
reserved	**1**	**bslbf**
mae_groupID	**7**	**uimsbf**
} else if (drcInstructionsType == 3) {		
reserved	**3**	**bslbf**
mae_groupPresetID	**5**	**uimsbf**
}		
reserved	**2**	**bslbf**
drcSetId	**6**	**uimsbf**
reserved	**1**	**bslbf**

Syntax	No of bits	Mnemonic
downmixId	**7**	**uimsbf**
reserved	**3**	**bslbf**
additionalDownmixIdCount	**3**	**uimsbf**
limiterPeakTargetPresent	**1**	**bslbf**
drcSetTargetLoudnessPresent	**1**	**bslbf**
for (j=0; j<additionalDownmixIdCount; j++) {		
reserved	**1**	**bslbf**
additionalDownmixId	**7**	**uimsbf**
}		
drcSetEffect	**16**	**uimsbf**
if (limiterPeakTargetPresent) {		
bsLimiterPeakTarget	**8**	**uimsbf**
}		
if (drcSetTargetLoudnessPresent) {		
reserved	**1**	**bslbf**
bsDrcSetTargetLoudnessValueUpper	**6**	**uimsbf**
drcSetTargetLoudnessValueLowerPresent	**1**	**bslbf**
if(drcSetTargetLoudnessValueLowerPresent) {		
reserved	**2**	**bslbf**
bsDrcSetTargetLoudnessValueLower	**6**	**uimsbf**
}		
}		
reserved	**1**	**bslbf**
dependsOnDrcSet	**6**	**uimsbf**
if (dependsOnDrcSet == 0) {		
noIndependentUse	**1**	**bslbf**
} else {		
reserved	**1**	**bslbf**
}		
}		
for (i=0; i<loudnessInfoCount; i++) {		
reserved	**6**	**bslbf**
loudnessInfoType	**2**	**uimsbf**
if (loudnessInfoType == 1 || loudnessInfoType == 2) {		
reserved	**1**	**bslbf**
mae_groupID	**7**	**uimsbf**
} else if (loudnessInfoType == 3) {		
reserved	**3**	**bslbf**
mae_groupPresetID	**5**	**uimsbf**
}		
loudnessInfo_size	**8**	**uimsbf**
loudnessInfo()		
}		
for (i=0; i<downmixIdCount; i++) {		
reserved	**1**	**bslbf**
downmixId	**7**	**uimsbf**
downmixType	**2**	**uimsbf**
CICPspeakerLayoutIdx	**6**	**uimsbf**
}		
}		
for (i=0; i<N; i++) {		
reserved	**8**	**bslbf**
}		
}		

2.6.117 Semantic definition of fields in MPEG-H 3D audio DRC and Loudness descriptor

mpegh3daDrcAndLoudnessInfoPresent – A one-bit flag signalling the presence of dynamic range control and loudness information in this descriptor.

drcInstructionsUniDrcCount – This field signals the number of DRC sets in the stream. This field can take values between 0 and 63, resulting in a maximum number of 63 DRC sets.

loudnessInfoCount – This field signals the number of loudness info blocks in the stream. This field can take values between 0 and 63, resulting in a maximum number of 63 loudness info blocks.

downmixIdCount – This field signals the number of downmixId definitions in the stream. This field can take values between 0 and 31, resulting in a maximum number of 31 downmixId definitions.

drcInstructionsType – See ISO/IEC 23008-3.

mae_groupID – See ISO/IEC 23008-3.

mae_groupPresetID – See ISO/IEC 23008-3.

drcSetId – See ISO/IEC 23003-4.

downmixId – See ISO/IEC 23003-4 and ISO/IEC 23008-3.

additionalDownmixIdCount – See ISO/IEC 23003-4.

limiterPeakTargetPresent – See ISO/IEC 23003-4.

drcSetTargetLoudnessPresent – See ISO/IEC 23003-4.

additionalDownmixId – See ISO/IEC 23003-4.

drcSetEffect – See ISO/IEC 23003-4.

bsLimiterPeakTarget – See ISO/IEC 23003-4.

bsDrcSetTargetLoudnessValueUpper – See ISO/IEC 23003-4.

drcSetTargetLoudnessValueLowerPresent – See ISO/IEC 23003-4.

bsDrcSetTargetLoudnessValueLower – See ISO/IEC 23003-4.

dependsOnDrcSet – See ISO/IEC 23003-4.

noIndependentUse – See ISO/IEC 23003-4.

loudnessInfoType – See ISO/IEC 23008-3.

loudnessInfo_size – The number of bytes of the immediately following loudnessInfo().

loudnessInfo() – One loudnessInfo() structure as defined in ISO/IEC 23003-4.

downmixType – See ISO/IEC 23008-3.

CICPspeakerLayoutIdx – See ISO/IEC 23008-3.

Data fields provided both in this descriptor and as in-band information in the IS/IEC 23008-3 stream shall be set to the same value.

2.6.118 MPEG-H 3D audio command descriptor

The MPEG-H 3D audio command descriptor encapsulates an MHAS packet of the type PACTYP_USERINTERACTION that contains an interaction command or type PACTYP_AUDIOSCENEINFO that contains the audio scene information. Both types are supported by a MPEG-H 3D audio decoder.

By inserting the MHAS packet contained in this descriptor into the MHAS elementary stream, a receiver can send this command to the MPEG-H 3D audio decoder. As the descriptor contains a complete MHAS packet including MHAS header, nothing else needs to be added when multiplexing these bytes, consecutively, on an MHAS packet boundary in the MHAS stream.

Table 2-125 – MPEG-H 3D audio command descriptor

Syntax	No of bits	Mnemonic
MPEG-H_3dAudio_command_descriptor() {		
for (i = 0; i < N; i++) {		
data	**8**	**bslbf**
}		
}		

data – Data bytes shall be contiguous bytes of data from a complete MHAS packet (including the MHAS header) of the type PACTYP_USERINTERACTION.

2.6.119 Quality extension descriptor

The quality extension descriptor shall be sent once per event or program and hence is signalled using a descriptor in the program map table. This descriptor shall appear in the elementary stream loop of the PID for which quality information is provided.

Dynamic quality metadata is stored in access units and is associated with one or more video frames. These access units are encapsulated in MPEG sections identified by stream_type value of 0x2F.

The quality extension descriptor describes metrics that are present in each Quality Access Unit, and the constant field size that is used for the values. The quality metrics are defined in 4.3 of ISO/IEC 23001-10.

The syntax of quality extension descriptor containing static metadata is shown in Table 2-126.

Table 2-126 – Quality extension descriptor

Syntax	No. bits	Mnemonic
Quality_extension_descriptor() {		
field_size_bytes	**8**	**uimsbf**
metric_count	**8**	**uimsbf**
for (i=0; i < metric_count; i++) {		
metric_code[i]	**32**	**uimsbf**
}		
}		

field_size_bytes – The constant size in byte of the value for a metric in each Quality Access Unit.

metric_count – The number of metrics for quality values in each Quality Access Unit

metric_code – The code name of the metrics in the Quality Access Unit.

Semantics for all the syntax elements above are specified in 4.2 of ISO/IEC 23001-10. The quality metrics to be signalled can also be found in 4.3 of ISO/IEC 23001-10 and include:

- PSNR = Peak Signal to Noise Ratio
- SSIM = Structural Similarity Index
- MS-SSIM = Multi-Scale Structural Similarity Index
- VQM = Video Quality Metric
- PEVQ = Perceptual Evaluation of Video Quality
- MOS = Mean Opinion Score
- FSIG = Frame significance

2.6.120 Virtual segmentation descriptor

The virtual segmentation descriptor (see Table 2-127) appears in the elementary stream descriptor loop in the PMT and is used to indicate that the current elementary stream is virtually segmented using boundary descriptors (see U.3.11). This segmentation may come in a set of partitions – e.g., one partition demarcates the stream into 10-s virtual segments, while another creates 2-s virtual segments. If the boundary descriptor carried in transport stream packets appears in the elemental stream declared by the PMT stream description carrying the virtual segmentation descriptor, it is an explicit indication of segment boundary point, otherwise a reference PID shall be defined in the virtual segmentation descriptor; that reference PID indicates that segment boundary points are indicated in another elemental stream, for which the boundary descriptor shall be present in the PMT. The virtual segmentation for elementary streams for which no virtual segmentation descriptor is present in the PMT is undefined.

Table 2-127 – Virtual segmentation descriptor

Syntax	No. bits	Mnemonic
Virtual_segmentation_descriptor(){		
if (descriptor_length > 1) {		
num_partitions	**3**	**uimsbf**
timescale_flag	**1**	**bslbf**
reserved	**4**	**bslbf**
if (timescale_flag == 1) {		
ticks_per_second	**21**	**uismbf**
maximum_duration_length_minus_1 (MDL)	**2**	**uismbf**
reserved	**1**	**bslbf**
}		
for (i = 0; i < num_partitions; i++) {		
explicit_boundary_flag	**1**	**bslbf**
partition_id	**3**	**uimsbf**
reserved	**4**	**bslbf**
SAP_type_max	**3**	**uismbf**
if (explicit_boundary_flag == 0) {		
reserved	**5**	**bslbf**
boundary_PID	**13**	**uimsbf**
reserved	**3**	**bslbf**
}		
else {		
maximum_duration	**MDL*8 + 5**	**uimsbf**
}		
}		
}		
}		

2.6.121 Semantic definition of fields in virtual segmentation descriptor

timescale_flag: If set to '1', timescale information is present. If set to '0', ticks_per_second is inferred to be 1, and MDL is inferred to be 0 (i.e., maximum_duration_length_minus_1=−1). The value of '0' allows maximum segment duration of up to 31 seconds, expressed in integer seconds.

ticks_per_second: Precision, in ticks per second, of the maximum_duration field, e.g., 0.1 second precision is 10 ticks/s, 0.01 second precision with 100 ticks/s, etc.

maximum_duration_length_minus_1: Length, in bytes (minus one), of the maximum_duration field variable byte length. This provides additional bytes in addition to the 5 bits pre-allocated to the maximum_duration field.

num_partitions: Number of partitions described in the virtual segmentation descriptor.

explicit_boundary_flag: If set to '0', this elementary stream is a dependent stream, and boundary data for it is provided on a reference partition on a different PID, specified by boundary_PID; otherwise, the current PID carries boundary descriptors.

partition_id: ID of the partition described in the boundary descriptor.

boundary_PID: PID carrying boundary_descriptor() that is used by this partition of this elementary stream.

SAP_type_max: Maximum possible value of SAP in this partition. If SAP_type_max value is 0, any SAP value may appear in the stream.

maximum_duration: Maximum virtual segment duration for a segment on partition partition_id, expressed in units of ticks_per_second. For consecutive virtual segments S(i) and S(i+1) on the above partition, if PTS(i) stands for the earliest PTS in segment S(i), and PTS(i+1) stands for the earliest PTS in segment S(i+1) this duration equals (PTS(i+1) – PTS(i))*ticks_per_second/90000. If set to 0, virtual segment duration is unlimited.

2.7 Restrictions on the multiplexed stream semantics

2.7.1 Frequency of coding the system clock reference

The program stream shall be constructed such that the time interval between the bytes containing the last bit of system_clock_reference_base fields in successive packs shall be less than or equal to 0.7 s. Thus:

$$|t(i) - t(i')| \leq 0.7\text{s}$$

for all i and i′ where i and i′ are the indexes of the bytes containing the last bit of consecutive system_clock_reference_base fields.

2.7.2 Frequency of coding the program clock reference

The transport stream shall be constructed such that the time interval between the bytes containing the last bit of program_clock_reference_base fields in successive occurrences of the PCRs in transport stream packets of the PCR_PID for each program shall be less than or equal to 0.1 s. Thus:

$$|t(i) - t(i')| \leq 0.1\text{s}$$

for all i and i′ where i and i′ are the indexes of the bytes containing the last bit of consecutive program_clock_reference_base fields in the transport stream packets of the PCR_PID for each program.

There shall be at least two (2) PCRs, from the specified PCR_PID within a transport stream, between consecutive PCR discontinuities (refer to 2.4.3.4) to facilitate phase locking and extrapolation of byte delivery times.

2.7.3 Frequency of coding the elementary stream clock reference

The program stream and transport stream shall be constructed such that if the elementary stream clock reference field is coded in any PES packets containing data of a given elementary stream the time interval in the PES_STD between the bytes containing the last bit of successive ESCR_base fields shall be less than or equal to 0.7 s. In PES Streams the ESCR encoding is required with the same interval. Thus:

$$|t(i) - t(i')| \leq 0.7\text{s}$$

for all i and i′ where i and i′ are the indexes of the bytes containing the last bits of consecutive ESCR_base fields.

NOTE – The coding of elementary stream clock reference fields is optional; they need not be coded. However, if they are coded, this constraint applies.

2.7.4 Frequency of presentation timestamp coding

The program stream and transport stream shall be constructed so that the maximum difference between coded presentation timestamps referring to each elementary video or audio stream is 0.7 s. Thus:

$$|tp_n(k) - tp_n(k'')| \leq 0.7\text{s}$$

for all n, k, and k″ satisfying:

- $P_n(k)$ and $P_n(k'')$ are presentation units for which presentation timestamps are coded;
- k and k″ are chosen so that there is no presentation unit, $P_n(k')$ with a coded presentation timestamp and with $k < k' < k''$; and
- No decoding discontinuity exists in elementary stream n between $P_n(k)$ and $P_n(k'')$.

The 0.7-s constraint does not apply in the case of:

- still pictures as defined in 2.1;
- AVC still pictures;
- AVC access units with a very low frame rate, where the presentation time of subsequent access units differs by more than 0.7 s. In this particular case, the VUI parameters num_units_in_tick and time_scale shall be present either in the AVC video stream or in an AVC-timing and HRD descriptor associated with the AVC video stream.

NOTE – The presentation time of an AVC access unit is equivalent to the DPB output time $t_{o,dpb}(n)$ defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

2.7.5 Conditional coding of timestamps

For each elementary stream of a program stream or transport stream, a presentation timestamp (PTS) shall be encoded for the first access unit.

A decoding discontinuity exists at the start of an access unit $A_n(j)$ in an elementary stream n if the decoding time $td_n(j)$ of that access unit is greater than the largest value permissible given the specified tolerance on the system_clock_frequency. For video, except when trick mode status is true or when low_delay flag is '1', this is allowed only at the start of a video sequence. If a decoding discontinuity exists in any elementary video or audio stream in the transport stream or program stream, then a PTS shall be encoded referring to the first access unit after each decoding discontinuity except when trick mode status is true.

When low_delay is '1' a PTS shall be encoded for the first access unit after an EB_n or B_n underflow.

A PTS may only be present in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 video or audio elementary stream PES packet header if the first byte of a picture start code or the first byte of an audio access unit is contained in the PES packet.

A decoding_timestamp (DTS) shall appear in a PES packet header if and only if the following two conditions are met:

- a PTS is present in the PES packet header;
- the decoding time differs from the presentation time.

For each AVC 24-hour picture, no explicit PTS and DTS value shall be encoded in the PES header. For such AVC access unit, decoders shall infer the presentation time from the parameters within the AVC video stream. Therefore, each AVC video stream that contains one or more AVC 24-hour picture(s):

- shall either carry picture timing SEI messages with coded values of cpb_removal_delay and dpb_output_delay; or
- shall carry VUI parameters with the fixed_frame_rate_flag set to '1' and shall carry Picture Order Count (POC) information (PicOrderCnt) whereby pictures are counted in units of $\Delta t_{fi,dpb}(n)$, where $\Delta t_{fi,dpb}(n)$ is specified in equation E-10 of Rec. ITU-T H.264 | ISO/IEC 14496-10.

  NOTE 1 – The requirements in the second bullet are met if an AVC timing and HRD descriptor is associated with the AVC video stream with the fixed_frame_rate_flag set to '1' and the temporal_poc_flag set to '1'.

The following applies to AVC access units in an AVC video stream carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream. For each AVC access unit that does not represent an AVC 24-hour picture, a PES header with a coded PTS and, if applicable, DTS value shall be provided, unless all conditions expressed under one of the following four bullets are true:

- In the AVC video sequence the following SEI messages are present, as signalled by VUI parameters:
  a) picture timing SEI messages providing the cpb_removal_delay and the dpb_output_delay parameters; and
  b) buffering period SEI messages providing the initial_cpb_removal_delay and the initial_cpb_removal_delay_offset parameters.

  NOTE 2 – When picture timing SEI messages are present in the AVC video sequence, then these messages are present for each AVC access unit, as required by Rec. ITU-T H.264 | ISO/IEC 14496-10. When buffering period SEI messages are present in the AVC video sequence, then these messages shall be present for each IDR access unit and for each access unit that is associated with a recovery point SEI message, as required by Rec. ITU-T H.264 | ISO/IEC 14496-10.
- An AVC timing and HRD descriptor is associated with the AVC video stream and in this descriptor the fixed_frame_rate_flag is set to '1' and the temporal_poc_flag is set to '1'.
- An AVC timing and HRD descriptor is associated with the AVC video stream and in this descriptor the fixed_frame_rate_flag is set to '1', the picture_to_display_conversion_flag is set to '1', the temporal_poc_flag is set to '0' and in the AVC video sequence picture timing SEI messages with the pic_struct field are present.

  NOTE 3 – In this specific case the pic_struct field is used to determine subsequent PTS values.
- An AVC timing and HRD descriptor is associated with the AVC video stream and in this descriptor the fixed_frame_rate_flag is set to '1' and the temporal_poc_flag is set to '0' and the picture_to_display_conversion_flag is set to '0'.

  NOTE 4 – In this case the POC information in the AVC video stream is used to determine the subsequent PTS values.

2.7.6 Timing constraints for scalable coding

If an audio sequence is coded using an extension bitstream, such as specified in ISO/IEC 13818-3, then corresponding decoding/presentation units in the two layers shall have identical PTS values.

If a video sequence is coded as an SNR enhancement of another sequence, such as specified in 7.8 of Rec. ITU-T H.262 | ISO/IEC 13818-2, then the set of presentation times for both sequences shall be the same.

If a video sequence is coded as two partitions, such as specified in 7.10 of Rec. ITU-T H.262 | ISO/IEC 13818-2, then the set of presentation times for both partitions shall be the same.

If a video sequence is coded as a spatial scalable enhancement of another sequence, such as specified in 7.7 of Rec. ITU-T H.262 | ISO/IEC 13818-2, then the following shall apply:

- If both sequences have the same frame rate, the set of presentation times for both sequences shall be the same.

  NOTE – This does not imply that the picture coding type is the same in both layers.
- If the sequences have different frame rates, the set of presentation times shall be such that as many presentation times as possible shall be common to both sequences.
- The picture from which the spatial prediction is made shall be one of the following:
  - the coincident or most recently decoded lower layer picture;
  - the coincident or most recently decoded lower layer picture that is an I- or P-picture;
  - the second most recently decoded lower layer picture that is an I- or P-picture, and provided that the lower layer does not have the low_delay flag set to '1'.

If a video sequence is coded as a temporally scalable enhancement of another sequence, such as specified in 7.9 of Rec. ITU-T H.262 | ISO/IEC 13818-2, then the following lower layer pictures may be used as the reference. Times are relative to presentation times of:

- the coincident or most recently presented lower layer picture;
- the next lower layer picture to be presented.

For AVC video streams conforming to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, there are no timing constraints on SVC dependency representations or re-assembled AVC access units of video sub-bitstreams of an AVC video stream.

2.7.7 Frequency of coding P-STD_buffer_size in PES packet headers

In a program stream, the P-STD_buffer_scale and P-STD_buffer_size fields shall occur in the first PES packet of each elementary stream and again whenever the value changes. They may also occur in any other PES packet.

2.7.8 Coding of system header in the program stream

In a program stream, the system header may be present in any pack, immediately following the pack header. The system header shall be present in the first pack of a program stream. The values encoded in all the system headers in the program stream shall be identical.

2.7.9 Constrained system parameter program stream

A program stream is a "Constrained System Parameters Stream" (CSPS) if it conforms to the bounds specified in this subclause. Program streams are not limited to the bounds specified by the CSPS. A CSPS may be identified by means of the CSPS_flag defined in the system header in 2.5.3.5. The CSPS is a subset of all possible program streams.

Packet rate

In the CSPS, the maximum rate at which packets shall arrive at the input to the P-STD is 300 packets per second if the value encoded in the rate_bound field (refer to 2.5.3.6) is less than or equal to 4 500 000 bits/s if the packet_rate_restriction_flag is set to '1', and less than or equal to 2 000 000 bits/s if the packet_rate_restriction_flag is set to '0'. For higher bit rates the CSPS packet rate is bounded by a linear relation to the value encoded in the rate_bound field.

Specifically, for all packs p in the program stream when the packet_rate_restriction_flag (refer to 2.5.3.5) is set to a value of '1',

$$NP \le (t(i') - t(i)) \times 300 \times \max\left[1, \frac{R_{\max}}{4.5 \times 10^6}\right] \qquad (2\text{-}27)$$

and if the packet_rate_restriction_flag is set to a value of '0'

$$NP \le (t(i') - t(i)) \times 300 \times \max\left[1, \frac{R_{\max}}{2.5 \times 10^6}\right] \qquad (2\text{-}28)$$

where:

$$R_{\max} = 8 \times 50 \times rate\_bound \quad \text{bit/s} \qquad (2\text{-}29)$$

NP is the number of packet_start_code_prefixes and system_header_start_codes between adjacent pack_start_codes or between the last pack_start_code and the MPEG_program_end_code as defined in Table 2-37 and semantics in 2.5.3.2.

t(i) is the time, measured in seconds, encoded in the SCR of pack p.

t(i′) is the time, measured in seconds, encoded in the SCR for pack p + 1, immediately following pack p, or in the case of the final pack in the program stream, the time of arrival of the byte containing the last bit of the MPEG_program_end_code.

Decoder buffer size

In the case of a CSPS the maximum size of each input buffer in the system target decoder is bounded. Different bounds apply for video elementary streams and audio elementary streams.

In the case of a Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 video elementary stream in a CSPS, the following applies:

BS_n has a size which is equal to the sum of the size of the Video Buffer Verifier (VBV) as specified in the Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 stream, respectively, and an additional amount of buffering BS_{add}. BS_{add} is specified as:

$$BS_{add} \le MAX\ [6 \times 1024,\ R_{vmax} \times 0.001] \text{ bytes}$$

where R_{vmax} is the maximum bit rate of the Rec. ITU-T H.262 | ISO/IEC 13818-2 or ISO/IEC 11172-2 video elementary stream.

In the case of a Rec. ITU-T H.264 | ISO/IEC 14496-10 video elementary stream in a CSPS, the following applies:

BS_n has a size which is equal to the sum of cpb_size and an additional amount of buffering BS_{add}. BS_{add} is specified as:

$$BS_{add} \le MAX\ [6 \times 1024,\ R_{vmax} \times 0.001] \text{ bytes}$$

where R_{vmax} is the maximum video bit rate of the AVC video stream, and

where cpb_size is the CpbSize[cpt_cnt_minus1] size of the CPB for the byte stream format signalled in the NAL hrd_parameters() in the AVC video stream. If the NAL hrd_parameters() are not present in the AVC video stream, then the cpb_size shall be the size defined as 1200 × MaxCPB in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 for the applied level.

In the case of an audio elementary stream in a CSPS, unless otherwise specified below, the following applies:

$$BS_n \le 4096 \text{ bytes}$$

In the case of ISO/IEC 13818-7 ADTS audio elementary stream in a CSPS the following applies to support 8 channels:

$$BS_n \le 8976 \text{ bytes}$$

2.7.10 Transport stream

Sample rate locking in transport streams

In the transport stream there shall be a specified constant rational relationship between the audio sampling rate and the system clock frequency in the system target decoder, and likewise a specified rational relationship between the video frame rate and the system clock frequency. The system_clock_frequency is defined in 2.4.2.1. The video frame rate is specified in Rec. ITU-T H.262 | ISO/IEC 13818-2 or in ISO/IEC 11172-2. The audio sampling rate is specified in ISO/IEC 13818-3 or in ISO/IEC 11172-3. For all presentation units in all audio elementary streams in the transport stream, the ratio of system_clock_frequency to the actual audio sampling rate, SCASR, is constant and equal to the value indicated in the following table at the nominal sampling rate indicated in the audio stream.

$$SCASR = \frac{system\_clock\_frequency}{audio\_sample\_rate\_in\_the\_T - STD} \quad (2\text{-}30)$$

The notation $\frac{X}{Y}$ denotes real division.

Nominal audio sampling frequency (kHz)	16	32	22.05	44.1	24	48
SCASR	27 000 000 / 16 000	27 000 000 / 32 000	27 000 000 / 22 050	27 000 000 / 44 100	27 000 000 / 24 000	27 000 000 / 48 000

For all presentation units in each ISO/IEC 11172-2 video and Rec. ITU-T H.262 | ISO/IEC 13818-2 video stream in the transport stream, the ratio of system_clock_frequency to the actual video frame rate, SCFR, is constant and equal to the value indicated in the following table at the nominal frame rate indicated in the video stream.

$$SCFR = \frac{system\_clock\_frequency}{frame\_rate\_in\_the\_T - STD} \quad (2\text{-}31)$$

Nominal frame rate (Hz)	23.976	24	25	29.97	30	50	59.94	60
SCFR	1 126 125	1 125 000	1 080 000	900 900	900 000	540 000	450 450	450 000

The values of the SCFR are exact. The actual frame rate differs slightly from the nominal rate in cases where the nominal rate is 23.976, 29.97, or 59.94 frames per second.

For ISO/IEC 14496-2 video streams carried in a transport stream, the time base of the ISO/IEC 14496-2 video stream, as defined by vop_time_increment_resolution, shall be locked to the STC and shall be exactly equal to N times system_clock_frequency divided by K, with N and K integers that have a fixed value within each visual object sequence, with K greater than or equal to N.

For Rec. ITU-T H.264 | ISO/IEC 14496-10 video streams, the time base of the Rec. ITU-T H.264 | ISO/IEC 14496-10 video stream shall be locked to the system clock frequency. The frequency of the AVC time base is defined by the AVC parameter time_scale, and this frequency shall be exactly equal to N times system_clock_frequency divided by K, with N and K integers that have a fixed value within each AVC video sequence and K greater than or equal to N. For example, if the time_scale is set to 90 000, then the frequency of the AVC time base is exactly equal to system_clock_frequency divided by 300.

2.8 Compatibility with ISO/IEC 11172

The program stream of this Recommendation | International Standard is defined to be forward compatible with ISO/IEC 11172-1. Decoders of the program stream as defined in this Recommendation | International Standard shall also support decoding of ISO/IEC 11172-1.

2.9 Registration of copyright identifiers

2.9.1 General

Parts 1, 2 and 3 of ISO/IEC 13818 provide support for the management of audiovisual works copyrighting. In Rec. ITU-T H.222.0 | ISO/IEC 13818-1 this is by means of a copyright descriptor, while Rec. ITU-T H.262 | ISO/IEC 13818-2 and ISO/IEC 13818-3 contain fields for identifying copyright holders through syntax fields in the elementary stream syntax. This Recommendation | International Standard presents the method of obtaining and registering copyright identifiers in Rec. ITU-T H.222.0 | ISO/IEC 13818-1.

Rec. ITU-T H.222.0 | ISO/IEC 13818-1 specifies a unique 32-bit copyright_identifier which is a work type code identifier (such as ISBN, ISSN, ISRC, etc.) carried in the copyright descriptor. The copyright_identifier enables identification of a wide number of Copyright Registration Authorities. Each Copyright Registration Authority may specify a syntax and semantic for identifying the audiovisual works or other copyrighted works within that particular copyright organization through appropriate use of the variable length additional_copyright_info field which contains the copyright number.

In the following subclause and Annexes L, M and N, the benefits and responsibilities of all parties to the registration of copyright_identifier are outlined.

2.9.2 Implementation of a Registration Authority (RA)

ISO/IEC JTC 1 shall call for nominations for an international organization which will serve as the Registration Authority for the **copyright_identifier** as defined in 2.6.24. The selected organization shall serve as the Registration Authority. The so-named Registration Authority shall execute its duties in compliance with Annex H/JTC 1 Directives. The registered copyright_identifier is hereafter referred to as the Registered Identifier (RID).

Upon selection of the Registration Authority, JTC 1 shall require the creation of a Registration Management Group (RMG) which will review appeals filed by organizations whose request for a RID to be used in conjunction with Rec. ITU-T H.222.0 | ISO/IEC 13818-1 has been denied by the Registration Authority.

Annexes L, M and N provide information on the procedure for registering a unique copyright identifier.

2.10 Registration of private data format

The registration descriptor of Rec. ITU-T H.222.0 | ISO/IEC 13818-1 is provided by this text in order to enable users of this Specification to unambiguously carry data when its format is not recognized by this Specification. This provision will permit this Specification to carry all types of data while providing for a method of unambiguous identification of the characteristics of the underlying private data.

2.10.1 General

In the following subclause and Annexes O and P, the benefits and responsibilities of all parties to the registration of private data format are outlined.

2.10.2 Implementation of a Registration Authority (RA)

ISO/IEC JTC 1/SC 29 shall call for nominations from member bodies of ISO or National Committees of IEC which will serve as the Registration Authority for the **format_identifier** as defined in 2.6.8 and 2.6.9. The selected organization shall serve as the Registration Authority. The so-named Registration Authority shall execute its duties in compliance with Annex H/JTC 1 Directives. The registered private data format_identifier is hereafter referred to as the Registered Identifier (RID).

Upon selection of the Registration Authority, JTC 1 shall require the creation of a Registration Management Group (RMG) which will review appeals filed by organizations whose request for an RID to be used in conjunction with this Specification has been denied by the Registration Authority.

Annexes O and P provide information on the procedures for registering a unique format identifier.

2.11 Carriage of ISO/IEC 14496 data

2.11.1 Introduction

A Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream may carry individual ISO/IEC 14496-2 and 14496-3 elementary streams as well as ISO/IEC 14496-1 audiovisual scenes with its associated streams. Typically, the ISO/IEC 14496 streams will be elements of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, as defined by the PMT in a transport stream and the PSM in a program stream.

For the carriage of ISO/IEC 14496 data in transport streams and program streams, distinction is made between individual elementary streams and an ISO/IEC 14496-1 audiovisual scene with its associated streams. For carriage of individual ISO/IEC 14496-2 and 14496-3 elementary streams, only system tools from Rec. ITU-T H.222.0 | ISO/IEC 13818-1 are used, as defined in 2.11.2. For carriage of an audiovisual ISO/IEC 14496-1 scene and associated ISO/IEC 14496 elementary streams, contained in ISO/IEC 14496-1 SL_packetized streams or FlexMux streams, tools from both Rec. ITU-T H.222.0 | ISO/IEC 13818-1 and from ISO/IEC 14496-1 are used, as defined in 2.11.3.

Carriage of Rec. ITU-T H.264 | ISO/IEC 14496-10 video over Rec. ITU-T H.222.0 | ISO/IEC 13818-1 streams is specified in 2.14.

Carriage of ISO/IEC 14496-17 text streams over Rec. ITU-T H.222.0 | ISO/IEC 13818-1 streams is specified in 2.15.

2.11.2 Carriage of individual ISO/IEC 14496-2 and 14496-3 Elementary Streams in PES packets

2.11.2.1 Introduction

Individual ISO/IEC 14496-2 and 14496-3 elementary streams may be carried in PES packets as PES_packet_data_bytes. For PES packetization no specific data alignment constraints apply. For synchronization PTSs and, when appropriate, DTSs are encoded in the header of the PES packet that carries the ISO/IEC 14496 elementary stream data; for PTS and DTS encoding the same constraints apply as for ISO/IEC 13818 elementary streams. See Table 2-128 for an overview of how to carry individual ISO/IEC 14496 streams within a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream.

Table 2-128 – Carriage of individual ISO/IEC 14496 streams in Rec. ITU-T H.222.0 | ISO/IEC 13818-1

ISO/IEC 14496-2 visual	Carriage in PES packets	Stream_type = 0x10	Stream_id = '1110 xxxx'
ISO/IEC 14496-3 audio	Carriage in PES packets	Stream_type = 0x11	Stream_id = '110x xxxx'

If a PTS or DTS is present in the PES packet header it shall refer to the visual object that follows either the first VOP start code or the first still texture object startcode that commences in the PES packet. Each ISO/IEC 14496-2 video stream carried by Rec. ITU-T H.222.0 | ISO/IEC 13818-1 shall contain the information required to decode the ISO/IEC 14496-2 video stream; consequently the stream shall contain Visual Object Sequence Headers, Visual Object Headers and Video Object Layer Headers.

In the case of an ISO/IEC 14496-3 elementary stream signalled by a stream_type value of 0x11, before PES packetization the elementary stream data shall be first encapsulated in the LATM transport syntax defined in ISO/IEC 14496-3. In such a case, if a PTS is present in the PES packet header it shall refer to the first audio frame that follows the first syncword that commences in the payload of the PES packet.

In the case of an ISO/IEC 14496-3 elementary stream signalled by a stream_type value of 0x1C, the first byte of each audio frame shall be the first byte of the payload of a PES packet; prior to PES packetization no encapsulation in any additional transport syntax shall be applied. An audio frame from an ISO/IEC 14496-3 elementary stream signalled by a stream_type value of 0x1C may be fragmented for carriage in multiple PES packets. In the PES packet header the data_alignment_indicator shall be set to '1' in each PES packet that carries a complete such audio frame or the first fragment thereof. The data_alignment_indicator shall be set to '0' for PES packets carrying subsequent (non-first) fragments of an audio frame.

Carriage of individual ISO/IEC 14496-2 and ISO/IEC 14496-3 elementary streams in PES packets shall be identified by appropriate stream_id and stream_type values, indicating the use of ISO/IEC 14496-2 Visual or 14496-3 Audio. In addition, such carriage shall be signalled by the MPEG-4_video descriptor or MPEG-4_audio descriptor, respectively. These descriptors shall be conveyed in the descriptor loop for the respective elementary stream entry in the Program Map Table in case of a transport stream or in the Program Stream Map, when present, in case of a program stream. Rec. ITU-T H.222.0 | ISO/IEC 13818-1 does not specify presentation of ISO/IEC 14496-2 and ISO/IEC 14496-3 elementary streams in the context of a program.

Carriage of an individual ISO/IEC 14496-17 text streams in PES packets shall be identified by appropriate stream_id and stream_type values, indicating the use of ISO/IEC 14496-17 text.

2.11.2.2 STD extensions for individual ISO/IEC 14496 elementary streams

The T-STD model includes a transport buffer TB_n and a multiplex buffer B_n prior to decoding of each individual ISO/IEC 14496 elementary stream n. Note that in the T-STD the single multiplex buffer B_n is also applied for ISO/IEC 14496-2 video, as indicated in Figure 2-4, instead of the approach with two buffers MB_n and EB_n used for ISO/IEC 13818-2 video in the T-STD. For buffers TB_n and B_n and the rate Rx_n between TB_n and B_n the following constraints apply.

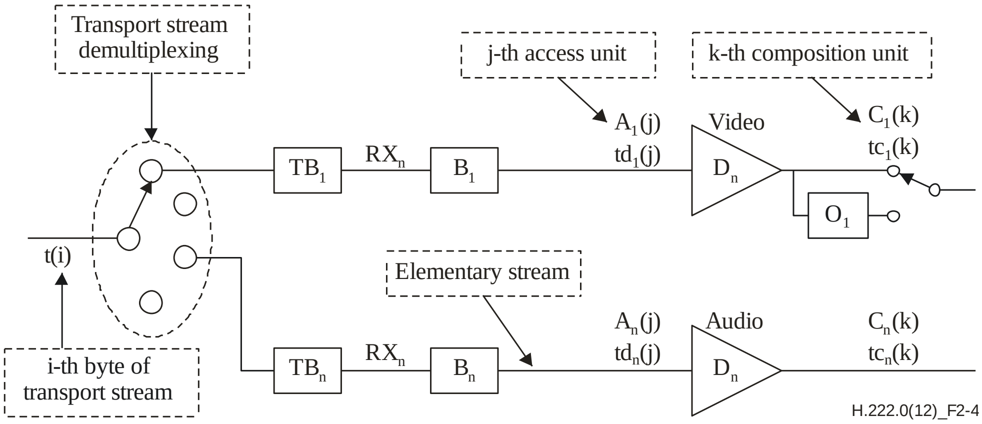


Figure 2-4 – T-STD model extensions for individual ISO/IEC 14496 elementary streams

In case of carriage of an ISO/IEC 14496-2 stream:

Size BS_n of Buffer B_n:

$$BS_n = BS_{mux} + BS_{oh} + VBV_{max}[\text{profile,level}]$$

where:

BS_{oh}, packet overhead buffering, is defined as:

BS_{oh} = (1/750) seconds × max{R_{max}[profile,level], 2 000 000 bit/s}

and:

BS_{mux}, additional multiplex buffering, is defined as:

BS_{mux} = 0.004 seconds × max{R_{max}[profile,level], 2 000 000 bit/s}

Rate Rx_n:

$$Rx_n = 1.2 \times R_{max}[\text{profile,level}]$$

where:

VBV_{max}[profile,level] and R_{max}[profile,level] are defined in ISO/IEC 14496-2 for each profile and level. For profiles and levels for which no VBV_{max} value is specified, the size of B_n and the rate Rx_n are user defined.

For carriage of an ISO/IEC 14496-3 audio stream the following applies.

Size BS_n of Buffer B_n, whereby $BS_n = BS_{mux} + BS_{dec} + BS_{oh}$:

For ISO/IEC 14496-3 audio, except for ISO/IEC 14496-3 DST, ALS and SLS:

BS_n = 3584 bytes if 1-2 channels

Here, the size of the access unit decoding buffer BS_{dec}, and the PES packet overhead buffer BS_{oh} are constrained by: $BS_{dec} + BS_{oh} \leq 2848$ bytes; a portion (736 bytes) of the 3584 byte buffer is allocated for buffering to allow multiplexing. The rest, 2848 bytes, are shared for access unit buffering BS_{dec}, BS_{oh} and additional multiplexing.

BS_n = 8976 bytes if 3-8 channels

BS_n = 12804 bytes if 9-12 channels

BS_n = 51216 bytes if 13-48 channels

For ISO/IEC 14496-3 DST-64, ALS and SLS audio:

if number of audio channels <= 8 then BS_n = 1 600 000 bytes,

else BS_n = 200 000 × (number of audio channels) bytes.

For ISO/IEC 14496-3 DST-128 audio:

BS_n = 400 000 × (number of audio channels) bytes.

For ISO/IEC 14496-3 DST-256 audio:

BS_n = 800 000 × (number of audio channels) bytes.

Rate Rx_n:

For ISO/IEC 14496-3 audio, except for ISO/IEC 14496-3 DST, ALS and SLS:

Rx_n = 2 000 000 bit/s if 1-2 channels

Rx_n = 5 529 600 bit/s if 3-8 channels

Rx_n = 8 294 400 bit/s if 9-12 channels

Rx_n = 33 177 600 bit/s if 13-48 channels

For ISO/IEC 14496-3 DST-64, ALS and SLS audio:

if number of audio channels <= 8 then Rx_n = 30 000 000 bit/s,

else Rx_n = 120 000 000 bit/s.

For ISO/IEC 14496-3 DST-128 and DST-256 audio:

Rx_n = 120 000 000 bit/s.

The P-STD model includes a multiplex buffer B_n prior to decoding of each individual ISO/IEC 14496 elementary stream n. The size BS_n of buffer B_n in the P-STD is defined by the P-STD_buffer_size field in the PES packet header.

2.11.3 Carriage of audiovisual ISO/IEC 14496-1 scenes and associated ISO/IEC 14496 streams

2.11.3.1 Introduction

This clause describes the encapsulation and signalling when an audiovisual scene represented by ISO/IEC 14496 data is carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program stream or transport stream. ISO/IEC 14496 content consists of the initial object descriptor and a variable number of streams such as object descriptor streams, scene description streams (carrying either BIFS-Command or BIFS-Anim access units), IPMP streams, OCI streams and audiovisual streams. Each of the ISO/IEC 14496 streams shall be contained in an SL-packetized stream and may optionally be multiplexed into a FlexMux stream, both defined in ISO/IEC 14496-1. For carriage in Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program stream or transport stream, these SL-packetized streams and FlexMux streams shall contain encoded Object Clock Reference (OCR) and FlexMux Clock Reference (FCR) fields as specified in 2.11.3.4 and in 2.11.3.5, respectively. The SL-packetized streams or FlexMux streams are then encapsulated either in PES packets or in ISO_IEC_14496_sections prior to transport stream packetization and multiplexing or program stream multiplexing. ISO_IEC_14496_sections are built on the long format of H.222.0 | ISO/IEC 13818-1 sections.

Additionally, an ISO/IEC 14496 audiovisual scene may refer to non SL-Packetized streams carried in an Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport stream using a "pid://PID_NUMBER" URL scheme instead of a "od://OD_ID" URL scheme.

ISO/IEC 14496 streams may derive their time base from the PCR of the program through the OCR_ES_ID mechanism.

2.11.3.2 Assignment of ES_ID values

An ISO/IEC 14496-1 scene carried over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream may associate a number of ISO/IEC 14496, ISO/IEC 13818 and other streams by the use of the ES_ID parameter. The scene and the associated streams may be carried over the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, but a scene may also reference streams carried elsewhere, for example over an IP network. How to identify such other means is not defined in this Specification.

ISO/IEC 14496-1 defines name scoping rules for identifiers. These rules allow the same ES_ID value to be used for two different streams within ISO/IEC 14496 content. When one or multiple ISO/IEC 14496-1 scenes are carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Program, duplicate ES_ID values shall not occur within the program such that each ISO/IEC 14496 SL-packetized stream or ISO/IEC 14496-1 FlexMux channel has a unique ES_ID value in the program.

2.11.3.3 Timing of ISO/IEC 14496 scenes and associated streams

When carried over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the object time base of each ISO/IEC 14496 stream shall be locked to the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 STC, that is:

$$\text{If } X(t) = f_{stc}(t)/f_{object}(t)$$

then the value of X(t) shall be constant at any time t.

where:

$f_{stc}(t)$ denotes the intended frequency of the STC at time t, i.e., 27 000 000 Hz

$f_{object}(t)$ denotes the frequency of the object time base at time t

The object time base of ISO/IEC 14496 streams carried over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream is conveyed as follows:

– The object time base of an SL-packetized stream carried in PES packets without the use of the FlexMux shall be conveyed by coded OCRs in the SL packet header of that stream. See 2.11.3.4.

– The object time base of SL-packetized streams carried in PES packets within a FlexMux stream shall be conveyed by FCRs in that FlexMux stream. See 2.11.3.5. Consequently, all ISO/IEC 14496 streams contained within the same FlexMux stream share the same object time base.

– The object time base of an SL-packetized stream carried in sections shall be conveyed by another ISO/IEC 14496 stream within the transport stream or program stream as indicated by the OCR_ES_ID field in the ES descriptor for that stream.

– The object time base of an SL-packetized stream whose OCR_ES_ID identifies a non SL-packetized stream with a PID equal to the PCR PID is fstc(t) / 300

The following constraints shall apply for encoding of OCRs and FCRs in SL-packetized streams and FlexMux streams carried over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream:

– The OCRs and FCRs in each SL-packetized stream and each FlexMux stream associated with the same scene shall have the same resolution.

– The resolution of OCRs and FCRs for a scene, f_{cr}, shall have a value smaller than or equal to 90 000 Hz.

– The ratio $(f_{stc}(t)/300)/f_{cr}$, shall be an integer value larger than or equal to one. Consequently the resolution of the OCR and FCR syntax elements may only take values such as 90 000 Hz, 45 000 Hz, 30 000 Hz, 22 500 Hz, 18 000 Hz, etc.

Within the above constraints and the ISO/IEC 14496-1 constraint that the resolution f_{cr} shall represent an integer number of cycles per second, f_{cr} can be selected as appropriate for the scene.

The ISO/IEC 14496 time stamps coded in the SL packet header shall refer to instants of the object time base of the stream carried in the SL packet. The resolution of each such time stamp shall be of a factor 2^k smaller than the resolution of the OCRs or FCRs associated with the stream, with k a positive integer larger than or equal to zero. To achieve the same wrap around, the length of the time stamp fields, TimeStampLength, shall be k bit smaller than the length of the OCR or FCR field, OCRLength and FCRLength, respectively. Hence for each stream the following conditions shall apply for encoding of time stamps:

– TimeStampResolution = (OCRResolution or FCRResolution respectively)/2^k, with k a positive integer larger than or equal to zero. ISO/IEC 14496-1 requires TimeStampResolution to represent an integer number of cycles per second.

– TimeStampLength = OCRLength or FCRLength respectively – k.

For SL-packetized streams inheriting their object time base from the PCR PID, the following considerations apply:

– TimeStampResolution = 90000 / 2k, with k a positive integer larger than or equal to zero.

– TimeStampLength = 33–k.

For SL-packetized streams carrying an OCR, the relationship between a value of the STC and the corresponding value of the object time base of a stream is established by associating PTS fields in PES packet headers with the OCR or FCR in SL packet headers and FlexMux Stream packets respectively, as specified in 2.11.3.6 and 2.11.3.7.

For SL-packetized streams inheriting their time base from the PCR, the object time base of such a stream is $f_{stc}(t)$ / 300.

2.11.3.4 Delivery timing of SL-packetized streams

To carry ISO/IEC 14496 content in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, ISO/IEC 14496-1 SL-packetized streams are used. In each SL-packetized stream carried in a PES packet without the use of FlexMux, the objectClockReference field shall be encoded as follows:

1) An objectClockReference (OCR) field shall be present in the first SL packet header of a SL-packetized stream.

2) The SL-packetized stream shall be constructed such that the time interval between the bytes containing the last bit of successive OCR fields shall be less than or equal to 0.7 s. Thus:

$$|t(i'') - t(i')| <= 0.7\,\mathrm{s}$$

for all i′ and i″ where i' and i″ are the indexes of the bytes containing the last bit of consecutive OCR fields in the SL-packetized stream.

If an objectClockReference is encoded in an SL packet header, also the instantBitrate field shall be coded.

2.11.3.5 Delivery timing of FlexMux streams

Next to SL-packetized streams also the ISO/IEC 14496-1 FlexMux tool may be used to carry ISO/IEC 14496 content in Rec. ITU-T H.222.0 | ISO/IEC 13818-1 streams. The payload of FlexMux packets shall consist of SL packets as specified in ISO/IEC 14496-1. In each FlexMux stream carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream the fmxClockReference field shall be encoded as follows:

1) An fmxClockReference (FCR) field shall be present in the first FlexMux packet of a FlexMux stream.
2) The FlexMux stream shall be constructed such that the time interval between the bytes containing the last bit of successive FCR fields shall be less than or equal to 0.7 s. Thus:

$$|t(i'') - t(i')| <= 0.7\,\mathrm{s}$$

for all i′ and i″ where i′ and i″ are the indexes of the bytes containing the last bit of consecutive FCR fields in the FlexMux stream.

3) All ISO/IEC 14496 time stamps within the SL-packetized streams carried within a FlexMux stream shall refer to instants of the object time base conveyed by the FCR fields in the FlexMux stream. The SL-packetized streams carried in FlexMux packets need not carry OCR fields. If OCR fields are present, they may be ignored.

2.11.3.6 Carriage of SL-packetized streams in PES packets

A single ISO/IEC 14496-1 SL-packetized stream may be mapped into a single PES stream. One and only one SL packet from an SL-packetized stream shall constitute the payload of one PES packet. PES packets that carry an SL-packetized stream shall be identified by stream_id = 0xFA in the PES packet header.

When an OCR field is coded in the SL packet header, a PTS shall be encoded in the header of the PES packet that carries such SL packet header. This PTS shall be encoded with the 33-bit value of the 90-kHz portion of the STC that corresponds to the value of the object time base at the instant in time indicated by the OCR.

The ES_ID associated with the SL-packetized stream shall be signalled by an SL descriptor as specified in 2.6.46.

2.11.3.7 Carriage of FlexMux streams in PES packets

PES packets with a payload consisting of FlexMux packets shall be identified by stream_id = 0xFB in the PES packet header. An integer number of FlexMux packets shall constitute the payload of one PES packet, i.e., the payload of a PES packet carrying a FlexMux stream shall start with a FlexMux packet header and shall end with the last byte of a FlexMux packet.

If an fmxClockReference (FCR) field is encoded in one of the FlexMux packets contained in a PES packet, then a PTS shall be encoded in the header of the PES packet that contains such FlexMux packet. This PTS shall be encoded with the 33-bit value of the 90-kHz portion of the STC that corresponds to the value of the object time base of the FlexMux stream at the instant in time indicated by the FCR. In case multiple FlexMux packets with an encoded FCR field are contained in a PES packet, the PTS shall correspond to the time indicated by the FCR in the first such FlexMux packet encountered in the payload of the PES packet.

The ES_IDs associated with each SL-packetized stream conveyed in the FlexMux stream shall be signalled by an FMC descriptor as specified in 2.6.44.

2.11.3.8 Carriage of SL packets and FlexMux packets in sections

For transport of ISO/IEC 14496 content in sections, ISO_IEC_14496_sections are defined. Any ISO/IEC 14496 stream may be carried over ISO_IEC_14496_sections. A single ISO_IEC_14496_section shall contain either an entire SL packet of an SL-packetized stream or an integer number of FlexMux packets each carrying an SL packet of the same ISO/IEC 14496-1 elementary stream.

Table 2-129 shows the syntax of ISO_IEC_14496_sections defined to convey ISO/IEC 14496-1 elementary streams, qualified by the table_id as either object descriptor stream data, scene description stream data or any other ISO/IEC 14496 stream data. Object descriptor stream data consists of an Object Descriptor Table that comprises a number of object descriptors. The Object Descriptor Table may be transmitted in multiple ISO_IEC_14496_sections. Scene description data consists of a Scene Description Table that may comprise a number of BIFS commands. The Scene Description Table may be transmitted in multiple ISO_IEC_14496_sections. It is not required that a complete table be received in order to process its payload. However, the payload of sections shall be processed in the correct order, as indicated by the value of the section_number field in the ISO_IEC_14496_section header bytes. Other ISO/IEC 14496 stream data consists of an ISO/IEC 14496 table. The ISO/IEC 14496 table may be transmitted in multiple ISO_IEC_14496_sections.

Table 2-129 – Section syntax for transport of ISO/IEC 14496 stream

Syntax	No. of bits	Mnemonic
ISO_IEC_14496_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
private_indicator	**1**	**bslbf**
reserved	**2**	**bslbf**
ISO_IEC_14496_section_length	**12**	**uimsbf**
table_id_extension	**16**	**uimsbf**
reserved	**2**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
if (PMT_has_SL_descriptor(current_PID)) {		
SL_Packet()		
}		
else if (PMT_has_FMC_descriptor(current_PID)) {		
for (i = 1; i < N1; i++)		
FlexMuxPacket()		
}		
else {		
for (i = 1; i < N2; i++)		
reserved	**8**	**bslbf**
}		
CRC_32	**32**	**rpchof**
}		

table_id – This 8-bit field shall be set to '0x04', '0x05', or '0x08', in case of an ISO_IEC_14496_section. A value of '0x04' indicates an ISO_IEC_14496_section that carries an ISO/IEC 14496-1 scene description stream. A value of '0x05' indicates an ISO_IEC_14496_section that carries an ISO/IEC 14496-1 object descriptor stream. A value of '0x08' indicates an ISO_IEC_14496_section that carries other ISO/IEC 14496 streams.

section_syntax_indicator – This 1-bit field shall be set to '1'.

private_indicator – This 1-bit field shall not be specified by this Specification.

ISO_IEC_14496_section_length – This 12-bit field shall specify the number of remaining bytes in the section immediately following the ISO_IEC_14496_section_length field up to the end of the ISO_IEC_14496_section. The value of this field shall not exceed 4093 (0xFFD).

table_id_extension – This 16-bit field shall not be specified by this Specification; its use and value are defined by the user.

version_number – This 5-bit field shall represent the version number of the Object Descriptor Table or Scene Description Table respectively. The version number shall be incremented by 1 modulo 32 with each new version of the table. Version control is at the discretion of the application.

current_next_indicator – This 1-bit field shall be set to 1.

section_number – This 8-bit field shall represent the number of the ISO_IEC_14496_section. The section_number field of the first ISO_IEC_14496_section of the Object Descriptor Table or the Scene Description Table shall have a value equal to 0x00. The value of section_number shall be incremented by 1 with each additional section in the table.

last_section_number – This 8-bit field shall specify the number of the last section of the Object Descriptor Table or Scene Description Table of which this section is a part.

PMT_has_SL_descriptor(current_PID) – A pseudo function that shall be true if an SL descriptor is contained in the descriptor loop in the Program Map Table for the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program element that conveys this ISO_IEC_14496_section.

SL_Packet() – A sync layer packet as specified in 10.2.2 of ISO/IEC 14496-1.

PMT_has_FMC_descriptor(current_PID) – A pseudo function that shall be true if an FMC descriptor is contained in the descriptor loop in the Program Map Table for the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program element that conveys this ISO_IEC_14496_section.

FlexMuxPacket() – A FlexMux packet as specified in 11.2.4 of ISO/IEC 14496-1.

CRC_32 – This 32-bit field shall contain the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire ISO_IEC_14496_section.

2.11.3.9 T-STD extensions

2.11.3.9.1 T-STD Model for 14496 content

Figure 2-5 shows extensions of the Transport System Target Decoder for delivery of ISO/IEC 14496 program elements encapsulated in Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport streams.

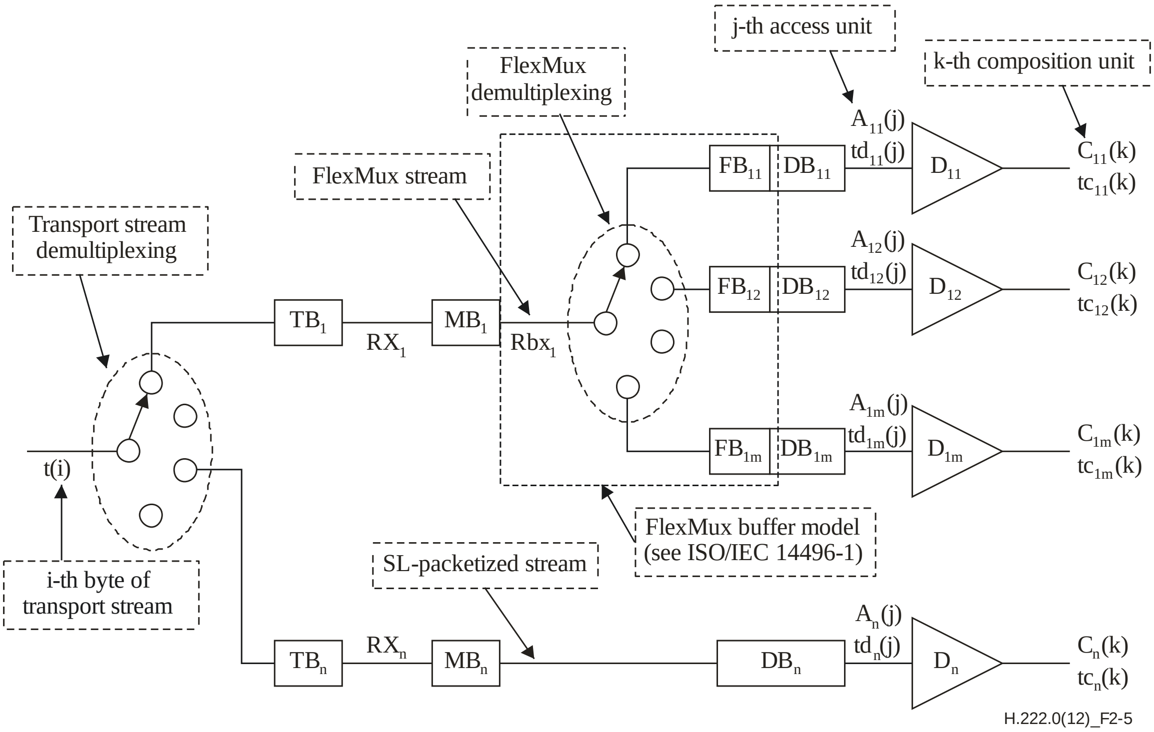


Figure 2-5 – T-STD model for ISO/IEC 14496 content

The following notation is used in Figure 2-5 and its description:

TB_n is the transport buffer.

MB_n is the multiplex buffer for FlexMux stream n or for SL-packetized stream n.

FB_{np} is the FlexMux buffer for the elementary stream in FlexMux channel p of FlexMux stream n.

DB_{np} is the decoder buffer for the elementary stream in FlexMux channel p of FlexMux stream n.

DB_n is the decoder buffer for elementary stream n.

D_{np} is the decoder for the elementary stream in FlexMux channel p of FlexMux stream n.

D_n is the decoder for elementary stream n.

Rx_n is the rate at which data are removed from TB_n.

Rbx_n is the rate at which data are removed from MB_n.

$A_{np}(j)$ is the jth access unit in elementary stream in FlexMux channel p of FlexMux stream n. $A_{np}(j)$ is indexed in decoding order.

$A_n(j)$ is the jth access unit in elementary stream n. $A_n(j)$ is indexed in decoding order.

$Td_{np}(j)$ is the decoding time, measured in seconds, in the system target decoder of the jth access unit in elementary stream in FlexMux channel p of FlexMux stream n.

$Td_n(j)$ is the decoding time, measured in seconds, in the system target decoder of the jth access unit in elementary stream n.

$C_{np}(k)$ is the kth composition unit in elementary stream in FlexMux channel p of FlexMux stream n. $C_{np}(k)$ results from decoding $A_{np}(j)$. $C_{np}(k)$ is indexed in composition order.

$C_n(k)$ is the kth composition unit in elementary stream n. $C_n(k)$ results from decoding $A_n(j)$. $C_n(k)$ is indexed in composition order.

$tc_{np}(k)$ is the composition time, measured in seconds, in the system target decoder of the kth composition unit in elementary stream in FlexMux channel p of FlexMux stream n.

$tc_n(k)$ is the composition time, measured in seconds, in the system target decoder of the kth composition unit in elementary stream n.

$t(i)$ indicates the time in seconds at which the ith byte of the transport stream enters the system target decoder.

2.11.3.9.2 Processing of FlexMux streams

Complete transport stream packets containing data from FlexMux stream n are passed to the transport buffer for FlexMux stream n, TB_n. The size of TB_n is fixed at 512 bytes. All bytes that enter TB_n are removed from TB_n at a rate Rx_n, specified by the TB_leak_rate field in the MultiplexBuffer descriptor associated with FlexMux stream n. When there is no data in buffer TB_n, rate Rx_n is equal to zero. Duplicate transport stream packets are not delivered to MB_n.

In case of carriage in PES packets, the PES packet header and payload data bytes are delivered to buffer MB_n; all other bytes leaving TB_n do not enter MB_n, and may be used to control the system. In case of carriage in ISO_IEC_14496_sections, the section header, payload and CRC-32 data bytes are delivered to buffer MB_n; all other bytes do not enter MB_n and may be used to control the system. In either case, the size of MB_n shall be specified by the MB_buffer_size field in the MultiplexBuffer descriptor.

The FlexMux Stream packet bytes in buffer MB_n are all delivered to their associated FlexMux buffer at the rate specified by the field fmxRate encoded in the FlexMux stream and in compliance with the FlexMux buffer model defined in 11.2.9 of ISO/IEC 14496-1. Only FlexMux packet payload data bytes in FlexMux channel p of FlexMux stream n enter buffer FB_{np}. FlexMux packet header bytes in FlexMux channel p of FlexMux stream n are discarded and may be used to control the system. The rate specified by the fmxRate field shall be applicable for all FlexMux packets in the stream immediately following the FlexMux Clock Reference channel packet up to the next encountered FlexMux Clock Reference channel packet. When there is no FlexMux stream data present in MB_n, no data is removed from MB_n. Bytes from the PES packet header or from the ISO_IEC_14496_section header that immediately precede a FlexMux header are instantaneously removed and discarded and may be used to control the system. Bytes from the ISO_IEC_14496_section CRC-32 fields that immediately follow the last FlexMux Stream packet in the section payload are removed instantaneously and discarded and may be used to verify the integrity of the data. Bytes from the FlexMux Clock Reference channel are instantaneously removed and discarded and may be used to lock the ISO/IEC 14496 object time base to the STC. When there is no PES packet or section payload data bytes, respectively present in MB_n, no data is removed from MB_n. All data that enters MB_n leaves it. All PES packet payload bytes of stream n enter the FlexMux demultiplexer instantaneously upon leaving MB_n.

2.11.3.9.3 Definition of FlexMux Buffer, FB_{np}

For each channel p of a FlexMux stream n, the size of FlexMux buffer FB_{np} is defined using the FmxBufferSize descriptor. FlexMux packet payload bytes are transferred from buffer FB_{np} to decoder buffer DB_{np} in compliance with the FlexMux buffer model defined in 11.2.9 of ISO/IEC 14496-1. Only SL packet payload bytes in FlexMux channel p of FlexMux stream n enter buffer DB_{np}. The SL packet header bytes in FlexMux channel p of FlexMux stream n are discarded and may be used to control the system.

2.11.3.9.4 Processing of SL-packetized streams

Complete transport stream packets containing data from SL-packetized stream n are passed to the transport buffer for SL-packetized stream n, TB_n. All bytes that enter TB_n are removed at a rate Rx_n, specified by the TB_leak_rate field in the MultiplexBuffer descriptor. When there is no data in buffer TB_n, rate Rx_n is equal to zero. Duplicate transport stream packets are not delivered to MB_n.

In case of carriage in PES packets, the PES packet header and payload data bytes are delivered to buffer MB_n; all other bytes leaving TB_n do not enter MB_n, and may be used to control the system. In case of carriage in ISO_IEC_14496_sections, the section header, payload and CRC-32 data bytes are delivered to buffer MB_n; all other bytes do not enter MB_n and may be used to control the system. In either case the size of MB_n is specified by the MB_buffer_size field in the MultiplexBuffer descriptor.

The SL-packetized stream bytes in buffer MB_n are all delivered to the decoder buffer DB_n at the rate specified by the field instantBitRate encoded in the SL-packetized stream and in compliance with the System Decoder Model defined in 7.4 of ISO/IEC 14496-1. The rate specified by the instantBitRate field shall be applicable for all data bytes in the SL-packetized stream immediately following the instantBitRate field in the SL packet header up to the next encountered instantBitRate field. If there are no SL-packetized stream bytes in MB_n, no bytes are removed from MB_n. Bytes from the PES packet header or from the ISO_IEC_14496_section header that immediately precede a SL packet header are instantaneously removed and discarded and may be used to control the system. Bytes from the ISO_IEC_14496_section CRC-32 fields that immediately follow the last SL packet payload byte in the section are removed instantaneously and discarded and may be used to verify the integrity of the data. When there are no PES packet or section payload data bytes, respectively present in MB_n, no data is removed from MB_n. All data that enters MB_n leaves it. All PES packet payload bytes of stream n enter buffer DB_n instantaneously upon leaving MB_n, with the exception of the SL packet headers. Bytes from the SL packet headers do not enter DB_n and may be used to control the system. The size of decoder buffer DB_n is given by the bufferSizeDB of the DecoderConfigDescriptor defined in ISO/IEC 14496-1.

2.11.3.9.5 Buffer management

Transport streams shall be constructed so that conditions defined in this subclause are satisfied.

TB_n shall not overflow and shall be empty at least once every second. MB_n shall not overflow. FB_{np} shall not overflow. DB_{np} and DB_n shall neither underflow nor overflow. Underflow of DB_{np} occurs when one or more bytes of an access unit are not present in DB_{np} at the decoding time associated with this access unit. Underflow of DB_n occurs when one or more bytes of an access unit are not present in DB_n at the decoding time associated with this access unit.

2.11.3.10 Carriage within a transport stream

2.11.3.10.1 Overview

A transport stream may contain one or more programs, each described by a Program Map Table. ISO/IEC 14496 content can be conveyed in addition to the already defined stream types for such a program. Elements of the ISO/IEC 14496 content may be conveyed in one or more Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program elements referenced by a unique PID value within a transport stream. As a special case, it is possible that a program within a transport stream consists only of ISO/IEC 14496 program elements. ISO/IEC 14496 content associated with a program and carried in the transport stream shall be referenced in the Program Map Table of that program. An initial object descriptor shall be used to define an ISO/IEC 14496-1 scene; the use of this descriptor is specified in 2.11.3.10.2.

Carriage of ISO/IEC 14496 content in a PID is signalled by a stream_type value of 0x12 or 0x13 in the Program Map Table in association with that PID value. A value of 0x12 indicates carriage in PES packets. The stream_id field in the PES packet header signals whether the PES packet contains a single SL packet or a number of FlexMux packets. A stream_type value of 0x13 in the Program Map Table indicates that the program element carries an object descriptor stream or a BIFS-Command stream contained in sections. In this case the table_id in the section header indicates whether an object descriptor stream is carried in the sections or a BIFS-Command stream. See also Table 2-130. The section contains either a single SL packet or a number of FlexMux packets, as indicated by the presence of an SL descriptor or a FMC descriptor respectively in the descriptor loop of the Program Map Table for the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program element that carries the sections. When ISO/IEC 14496 content is carried, the SL descriptor and the FMC descriptor shall specify the ES_ID for each encapsulated ISO/IEC 14496 stream. When the assignment of ES_ID values changes, the Program Map Table shall be updated and the version_number of the PMT shall be incremented by 1 modulo 32. An example of a content access procedure for ISO/IEC 14496 program components within a transport stream is given in Annex R.

Table 2-130 – ISO/IEC defined options for carriage of an ISO/IEC 14496 scene and associated streams in Rec. ITU-T H.222.0 | ISO/IEC 13818-1

ISO/IEC 14496-1 object descriptor streams	Encapsulation in SL packets	Carriage in PES packets	Stream_type = 0x12	Stream_id = '1111 1010'
		Carriage in ISO_IEC_ 14496_sections	Stream_type = 0x13	Table_id = 0x05
	Encapsulation in SL packets followed by Multiplex into FlexMux packets	Carriage in PES packets	Stream_type = 0x12	Stream_id = '1111 1011'
		Carriage in ISO_IEC_ 14496_sections	Stream_type = 0x13	Table_id = 0x05
ISO/IEC 14496-1 scene description streams	Encapsulation in SL packets	Carriage in PES packets	Stream_type = 0x12	Stream_id = '1111 1010'
		Carriage in ISO_IEC_ 14496_sections	Stream_type = 0x13	Table_id = 0x04
	Encapsulation in SL packets followed by Multiplex into FlexMux packets	Carriage in PES packets	Stream_type = 0x12	Stream_id = '1111 1011'
		Carriage in ISO_IEC_ 14496_sections	Stream_type = 0x13	Table_id = 0x04
All other ISO/IEC 14496 streams	Encapsulation in SL packets	Carriage in PES packets	Stream_type = 0x12	Stream_id = '1111 1010'
	Encapsulation in SL packets followed by Multiplex into FlexMux packets	Carriage in PES packets	Stream_type = 0x12	Stream_id = '1111 1011'

2.11.3.10.2 Initial Object Descriptor

In case of carriage of an ISO/IEC 14496-1 scene, the ISO/IEC 14496-1 initial object descriptor serves as the initial access point to all associated streams. The initial object descriptor shall be conveyed in the IOD descriptor located in the descriptor loop immediately following the program_info_length field in the Program Map Table of the program to which the scene is associated. It contains ES_Descriptors identifying the scene description and object descriptor streams that form part of this program. It may also contain ES_Descriptors identifying one or more associated IPMP or OCI streams. Identification of streams is done by means of ES_IDs as specified in clause 8 of ISO/IEC 14496-1.

2.11.3.11 P-STD Model for 14496 content

Figure 2-6 shows the STD model when ISO/IEC 14496 systems data are carried in a program stream.

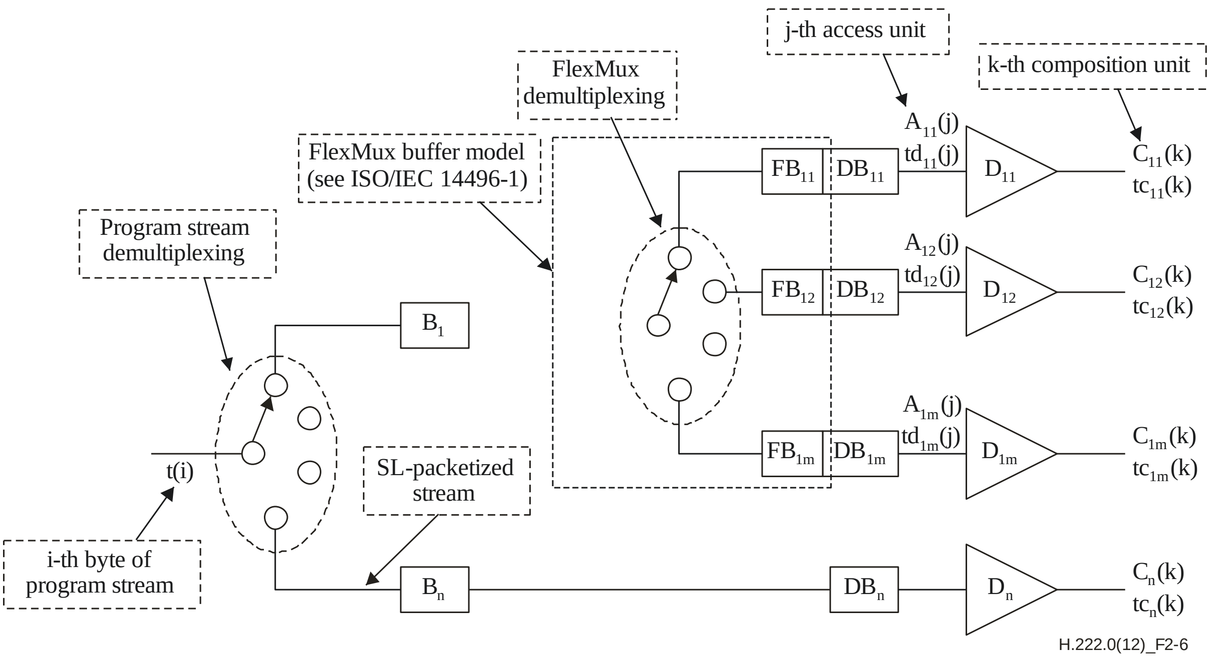


Figure 2-6 – P-STD model for ISO/IEC 14496 Systems stream

The following notation is used in Figure 2-6 and its description:

B_n is the input buffer for FlexMux stream n or for SL-packetized stream n.

FB_{np} is the FlexMux buffer for the elementary stream in FlexMux channel p of FlexMux stream n.

DB_{np} is the decoder buffer for the elementary stream in FlexMux channel p of FlexMux stream n.

DB_n is the decoder buffer for elementary stream n.

D_{np} is the decoder for elementary stream in FlexMux channel p of FlexMux stream n.

D_n is the decoder for elementary stream n.

$A_{np}(j)$ is the jth access unit in elementary stream in FlexMux channel p of FlexMux stream n. $A_{np}(j)$ is indexed in decoding order.

$A_n(j)$ is the jth access unit in elementary stream n. $A_n(j)$ is indexed in decoding order.

$Td_{np}(j)$ is the decoding time, measured in seconds, in the system target decoder of the jth access unit in elementary stream in FlexMux channel p of FlexMux stream n.

$Td_n(j)$ is the decoding time, measured in seconds, in the system target decoder of the jth access unit in elementary stream n.

$C_{np}(k)$ is the kth composition unit in elementary stream in FlexMux channel p of FlexMux stream n. $C_{np}(k)$ results from decoding $A_{np}(j)$. $C_{np}(k)$ is indexed in composition order.

$C_n(k)$ is the kth composition unit in elementary stream n. $C_n(k)$ results from decoding $A_n(j)$. $C_n(k)$ is indexed in composition order.

$tc_{np}(k)$ is the composition time, measured in seconds, in the system target decoder of the kth composition unit in elementary stream in FlexMux channel p of FlexMux stream n.

$tc_n(k)$ is the composition time, measured in seconds, in the system target decoder of the kth composition unit in elementary stream n.

$t(i)$ indicates the time in seconds at which the ith byte of the program stream enters the system target decoder.

2.11.3.11.1 Processing of FlexMux streams

At the input of the STD each byte in the payload of PES packets carrying a FlexMux stream n is transferred instantaneously to buffer B_n. The i-th byte enters B_n at time t(i). PES packet header bytes do not enter buffer B_n and may be used to control the system. The size of B_n is specified by the P-STD_buffer_size field in the header of the PES packet that carries stream n.

The FlexMux stream packet bytes in buffer B_n are all delivered to their associated FlexMux buffer at the rate specified by the field fmxRate encoded in the FlexMux stream and in compliance with the FlexMux buffer model defined in 11.2.9 of ISO/IEC 14496-1. Only FlexMux packet payload data bytes in FlexMux channel p of FlexMux stream n enter buffer FB_{np}. FlexMux packet header bytes in FlexMux channel p of FlexMux stream n are discarded and may be used to control the system. The rate specified by the fmxRate field shall be applicable for all FlexMux packets in the stream up to the next encountered FlexMux Clock Reference channel packet. Bytes from the FlexMux Clock Reference channel are instantaneously removed and discarded and may be used to lock the ISO/IEC 14496 object time base to the STC. When there is no PES packet payload data present in B_n, no data is removed from B_n. All data that enters B_n leaves it. All PES packet payload bytes of stream n enter the FlexMux demultiplexer instantaneously upon leaving B_n.

2.11.3.11.2 Definition of FlexMux Buffer, FB_{np}

For each channel p of a FlexMux stream n, the size of FlexMux buffer FB_{np} is defined using the FmxBufferSize descriptor if a Program Stream Map is present in the program stream. FlexMux packet payload bytes are transferred from buffer FB_{np} to decoder buffer DB_{np} in compliance with the FlexMux buffer model defined in 11.2.9 of ISO/IEC 14496-1. Only SL packet payload bytes in FlexMux channel p of FlexMux stream n enter buffer DB_{np}. The SL packet header bytes in FlexMux channel p of FlexMux stream n are discarded and may be used to control the system

2.11.3.11.3 Processing of SL-packetized streams

At the input of the STD each byte in the payload of PES packets carrying an SL-packetized stream n is transferred instantaneously to buffer B_n. The i-th byte enters B_n at time t(i). PES packet header bytes do not enter buffer B_n and may be used to control the system. The size of B_n is specified by the P-STD_buffer_size field in the header of the PES packet that carries stream n. The SL-packetized stream bytes in buffer B_n are delivered to the decoder buffer DB_n at the rate specified by the field instantBitRate encoded in the SL-packetized stream and in compliance with the System Decoder Model defined in 7.4 of ISO/IEC 14496-1. The rate specified by the instantBitRate field shall be applicable for all data bytes in the SL-packetized stream up to the next encountered instantBitRate field. When there is no PES packet payload data present in B_n, no data is removed from B_n. All data that enters B_n leaves it. All bytes of stream n enter buffer DB_n instantaneously upon leaving B_n, with the exception of the SL packet headers. Bytes from the SL packet headers do not enter DB_n and may be used to control the system. The size of decoder buffer DB_n is given by the bufferSizeDB of the DecoderConfigDescriptor defined in ISO/IEC 14496-1.

2.11.3.11.4 Buffer management

Program streams shall be constructed so that B_n does not overflow. FB_{np} shall not overflow. DB_{np} and DB_n shall neither underflow nor overflow. Underflow of DB_{np} occurs when one or more bytes of an access unit are not present in DB_{np} at the decoding time associated with this access unit. Underflow of DB_n occurs when one or more bytes of an access unit are not present in DB_n at the decoding time associated with this access unit.

2.11.3.12 Carriage within a program stream

2.11.3.12.1 Overview

A program stream contains only one program. ISO/IEC 14496 data can be conveyed in addition to the already defined stream types for such a program. As a special case, it is also possible that a program stream carries only ISO/IEC 14496 data. If a Program Stream Map is present, ISO/IEC 14496 content carried in the program stream shall be referenced as follows. Carriage of ISO/IEC 14496-1 scenes and associated ISO/IEC 14496 streams in SL and FlexMux packets is indicated by the appropriate stream_id and by an initial object descriptor; the use of this descriptor is specified in 2.11.3.12.2. For each carried ISO/IEC 14496 stream the SL descriptor and the FMC descriptor shall specify the ES_ID. When the assignment of ES_ID values changes, the Program Stream Map, if present, shall be updated and the program_stream_map_version shall be incremented by 1 modulo 32. Note that in a Program Stream the ISO/IEC 14496 content may also be referenced by private means.

For an example of a content access procedure for ISO/IEC 14496 program components within a program stream, see Annex R.

2.11.3.12.2 Initial object descriptor

In case of carriage of an ISO/IEC 14496-1 scene, the ISO/IEC 14496 initial object descriptor serves as the initial access point to all associated streams. If a Program Stream Map is present in the program stream, the initial object descriptor shall be conveyed in the IOD descriptor that is located in the descriptor loop immediately following the program_stream_info_length field. It contains ES_Descriptors identifying the scene description and object descriptor streams of the scene that form part of this program. It may also contain ES_Descriptors identifying one or more associated IPMP or OCI streams. Identification of streams is done by means of ES_IDs as specified in clause 8 of ISO/IEC 14496-1. In a program stream, the initial object descriptor may also be conveyed by private means.

2.12 Carriage of metadata

2.12.1 Introduction

A Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream can carry metadata. The format of the metadata may be defined by ISO or by any other authority. This subclause defines how to carry the metadata; transport mechanisms are defined as well as metadata related-signalling, the applied metadata timing model and extensions of the STD model for decoding of metadata.

A metadata service is defined to be a coherent set of metadata of the same format delivered to a receiver for a specific purpose. Metadata services are contained in metadata streams; each metadata stream carries one or more metadata services. This Specification assumes the notion of metadata Access Units within a metadata service. The definition of a Metadata Access Unit is metadata format specific, but each metadata service is assumed to represent a concatenation (or a collection) of metadata Access Units.

When transporting a metadata service over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, a unique metadata service id is assigned to each such service. A metadata service id references uniquely a metadata service among all the metadata services available on the same transport or program stream, and *not* unique *solely* within a metadata stream. The metadata service identifier is used to retrieve the metadata service and all the information needed to decode it.

Decoding of metadata may require the availability of decoder configuration data. If a metadata service carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream requires decoder configuration data for decoding, then this metadata decoder configuration data shall be carried within the same program of the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream.

Clause 2.12.2 discusses metadata timing, while 2.12.3 provides an overview of tools that are defined for transport of metadata over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream. The use of available transport tools is specified in 2.12.4 up to 2.12.8, and 2.12.9 specifies metadata related signalling. Finally, the STD model for metadata decoding is specified in 2.12.10.

Since many forms of metadata may be carried, it is essential to signal both the precise format and encoding of the metadata, and the semantic meaning the metadata conveys. The former is signalled by the metadata format, while the latter is signalled by the metadata application format. In other words, the metadata format conveys how the metadata shall be decoded, while the metadata application format conveys how to use the metadata, essentially which application uses the metadata. This division is important since it separates the encoding or representation of the metadata from its meaning, thereby allowing an application to be agnostic of the means by which its metadata is conveyed.

2.12.2 Metadata time-line model

Metadata may refer to time codes associated with the content, for example to indicate the beginning of a content segment. Each time indication made in the metadata refers to a certain metadata content time line specific to the actual metadata format and/or metadata application format. For example, one metadata (application) format may use UTC, while another metadata application format may use SMPTE time codes. To allow for transport of the content at any time over any media, the metadata content time line is expected but not required to be transport agnostic.

When transporting content and the associated metadata over Rec. ITU-T H.222.0 | ISO/IEC 13818-1 streams, accurate time references from the metadata to the content are to be maintained. The same is needed if the metadata is delivered over other means. To achieve this, the time line model of Figure 2-7 is assumed in this Specification.

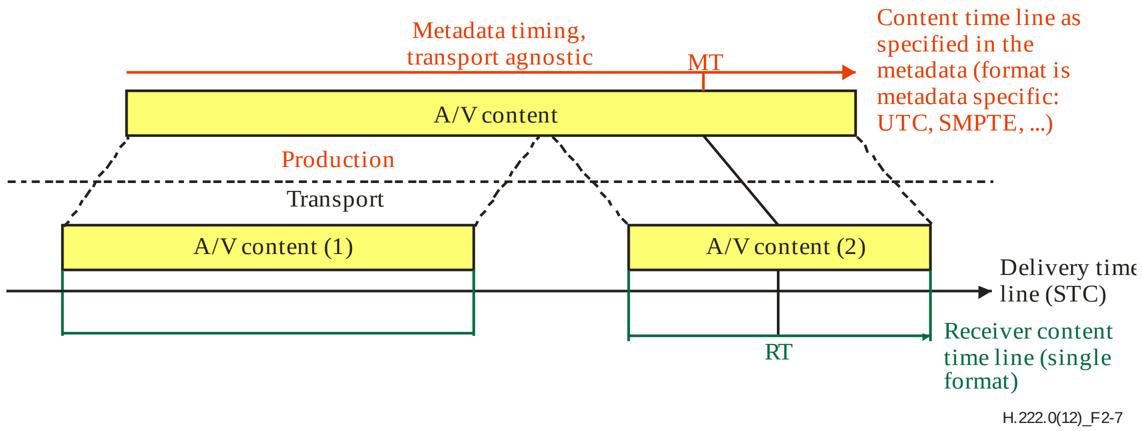


Figure 2-7 – Timing model for delivery of content and metadata

Metadata is associated with the audiovisual content, usually in a transport agnostic way, at production or any other stage prior to transport. Where needed, time information is embedded in the metadata to indicate for example specific segments within the content, using the metadata content time line used in the metadata. For example UTC or SMPTE time codes may be used. The time line format is independent of any time code that may or may not be embedded in the audiovisual stream itself. For example, the metadata time line may utilize UTC, while SMPTE time codes are embedded in the video stream.

The following requirements shall be met for each metadata stream:

- no time discontinuities shall occur in the metadata content time line;
- the metadata content time line shall be locked to the sampling clock of the content;
- each time reference in the metadata stream refers to the same metadata content time line.

At transport, a transport-specific timing is associated with the content; this is the delivery time line. In the case of transport over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the delivery time line is provided by the System Time Clock, the STC. The content may be delivered as a contiguous piece of information, but it is also possible to interrupt the delivery of the content, for example in the case of news-flash interruptions of a program; in such and other cases time line discontinuities may occur.

When time references are used in the metadata, in the System Target Decoder (STD) these time references are to be associated unambiguously with time values in the received content. To achieve this, a receiver content time line is required. The STC can be used as the receiver content time line, but due to STC discontinuities that may occur, the STC does not necessarily offer an unambiguous time association. Therefore the NPT (Normal Play Time) concept from ISO/IEC 13818-6 DSM-CC is also available for use as the receiver content time line. In any playback mode, such as normal, reverse, slow motion, fast forward, fast backward and still picture, the NPT provides an unambiguous time association, independent of STC discontinuities, and independent of insertions of other content. Note that a new NPT_reference_descriptor needs to be transmitted when the STC rolls over.

To maintain the accurate time references from metadata to the content, information is needed how to map a metadata time, MT, defined on the metadata content time line to the corresponding receiver time, RT, of the receiver content time line. This is achieved by providing the offset in time (in 90-kHz units) between the metadata content time line and the receiver content time line. The offset is provided in the content labelling descriptor. The offset conveys the value of the metadata time base at the instant in time at which the receiver content time base reaches a specified value. See also Figure 2-7.

The timing in metadata systems may refer to a specific picture or audio frame, for example using SMPTE time codes. The offset in time between the metadata content time line and the receiver content time line is expressed in units of 90 kHz, and consequently the metadata time reference will translate into a 90-kHz value in receivers. To accommodate for inaccuracies, receivers shall assume that when reference is made to a picture or audio frame the closest match shall be used. For example, the translated 90-kHz metadata time reference shall be matched with the picture or frame whose PTS value is closest to the translated value.

When using NPT, during playback in any mode at any point in time the offset remains constant between the metadata time base and the NPT time base. As long as neither STC discontinuities nor insertions with other content occur, the same is true for the offset in time between the metadata time base and the STC time base, but only in normal playback mode. For privately defined time lines the offset is also required to be constant, but possibly within constraints not defined in this Specification.

When synchronous transport of metadata is applied in PES packets or by using the synchronized DSM-CC download protocol, PTSs are assigned to the metadata. Such PTS may for example indicate the point in time at which the metadata becomes valid. This implies *a priori* knowledge of how to associate the metadata to the delivery timing. However, synchronously transported metadata may also contain time references, which are to be mapped from the metadata content time line to the receiver content time line using the specified offset between both time lines. See also Figure 2-8.

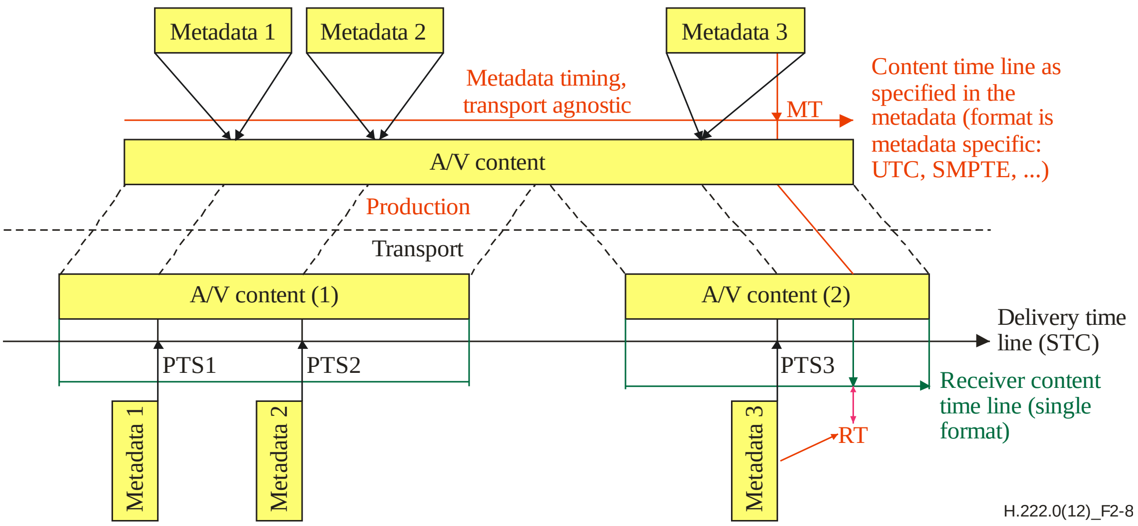


Figure 2-8 – Delivery of metadata in PES packets

2.12.3 Options for transport of metadata

To acknowledge the very diverse characteristics of metadata, a variety of tools is defined to transport the metadata over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream.

This Specification defines two tools for synchronous delivery of the metadata:

- carriage in PES packets;
- use of DSM-CC synchronized download protocol.

In addition, this Specification defines three tools for asynchronous delivery of metadata:

- carriage in metadata sections;
- use of DSM-CC data carousels;
- use of DSM-CC object carousels.

Note that some of the asynchronous transport options support carousels and file structures. The choice of transport tool depends on the requirements that apply to the delivery of the metadata, and the requirements of the tools, as described in the following subclauses.

Metadata may also be carried by private means such as PES packets with stream id value 0xBD or 0xBF (private_stream_id_1 or private_stream_id_2) or private sections. This Specification does not specify how to use private means for carriage of metadata, but allows for signalling of such metadata using the descriptors defined in 2.6.56 up to 2.6.63.

The basic referencing of metadata services is the same for all tools, using the metadata service id. However, there are differences per tool. When PES packets, metadata sections, or synchronized DSM-CC download sections are used, data from each metadata service is explicitly signalled within a metadata stream, using the metadata_service_id field. However, when using DSM-CC carousels, this signalling is left at the discretion of metadata applications. Note that this Specification allows for carriage of a metadata service in a DSM-CC carousel, but does not constrain how many metadata services can be carried in one DSM-CC carousel.

Metadata decoder configuration data is signalled explicitly when carried in a metadata descriptor, in PES packets with stream_type 0x15 and stream_id 0xFC, in metadata sections or in synchronized DSM-CC download sections. When metadata decoder configuration data is carried in a DSM-CC carousel, the signalling of such data is required, but not defined by this Specification; instead, such signalling is left at the discretion of applications.

2.12.4 Use of PES packets to transport metadata

2.12.4.1 General

PES packets provide a mechanism for synchronous transport of metadata. By means of the PTS in the PES packet header the metadata access units are associated with a certain instant of the STC, without the need for time references in the metadata. This implies *a priori* knowledge of how to associate the metadata to the delivery timing. Specific stream_id and stream_type values are assigned to signal PES packets carrying metadata; see 2.12.9.

When using PES packets with a stream_type of 0x15 and a stream_id of 0xFC to transport the metadata, a Metadata Access Unit Wrapper shall be used as the tool to align PES packets and the metadata Access Units, using metadata_AU_cells. This allows random access indication, whose meaning depends on the format of the metadata, and a cell sequence counter to identify loss of metadata_AU_cells. Each metadata Access Unit is carried and, if appropriate, fragmented in one or more metadata_AU_cells. In each PES packet that carries metadata, the first PES_packet_data_byte shall be the first byte of a Metadata_AU_cell. For each metadata Access Unit contained in the same PES packet, the PTS in the PES header applies. The PTS signals the time at which the metadata Access Units are decoded instantaneously and removed from buffer B_n in the STD. Note that the relationship between a decoded metadata Access Unit and audiovisual content is beyond the scope of this Specification.

A PES packet may contain a single metadata_AU_cell. This is useful if a metadata Access Unit does not fit into a single PES packet, in which case the fragmentation of the metadata Access Unit is handled by the metadata_AU_cell.

When metadata is carried by PES packets in a program stream, and if a Program Stream Map is applied in that program stream, then the Program Stream Map shall specify which PES packets contain the associated metadata.

2.12.4.2 Metadata Access Unit Wrapper

The metadata Access Unit Wrapper (see Table 2-131) shall be used when carrying metadata Access Units in PES packets with a stream_type of 0x15 and a stream_id value of 0xFC or in synchronized DSM-CC download sections of stream_type 0x19. The wrapper defines a structure consisting of a concatenated number of Metadata_AU_cells. By coding the size of the contained metadata in each metadata_AU_cell, metadata agnostic parsing is possible in receivers: the parser can retrieve the metadata and provide it to a metadata decoder without *a priori* knowledge on any detail of the metadata. The Metadata_AU_cell shall be aligned with the transport; that is the first byte of the payload of the PES packet or synchronized DSM-CC download section shall be the first byte of a Metadata_AU_cell.

If a metadata Access Unit does not fit entirely into a metadata_AU_cell, then the metadata Access Unit shall be fragmented into multiple metadata_AU_cells, where the fragmentation_indication in each such metadata_AU_cell signals that the metadata_AU_cell contains a fragment.

To each Metadata_AU_cell that is contained in the same PES packet or synchronized download section, the PTS as coded in the header of the PES packet or synchronized download section, respectively, applies.

Table 2-131 – Metadata Access Unit Wrapper

Syntax	No. of bits	Mnemonic
Metadata_AU_wrapper () {		
for (i = 0; i < N; i++){		
Metadata_AU_cell ()		
}		
}		

Table 2-132 – Metadata AU cell

Syntax	No. of bits	Mnemonic
Metadata_AU_cell () {		
metadata_service_id	**8**	**uimsbf**
sequence_number	**8**	**uimsbf**
cell_fragment_indication	**2**	**bslbf**
decoder_config_flag	**1**	**bslbf**
random_access_indicator	**1**	**bslbf**
reserved	**4**	**bslbf**
AU_cell_data_length	**16**	**uimsbf**
for (i = 0; I < AU_cell_data_length; i++){		
AU_cell_data_byte	**8**	**bslbf**
}		
}		

metadata_service_id: This 8-bit field identifies the metadata service associated with the metadata Access Unit carried in this metadata AU cell.

sequence_number: This 8-bit field specifies the sequence number of the metadata_AU_cell. This number increments by one for each successive metadata_AU_cell constituting the metadata_AU_wrapper, independent of the coded value of the metadata_service_id.

cell_fragment_indication: This 2-bit field conveys information on the metadata Access Unit carried in this metadata_AU_cell, corresponding to Table 2-133.

Table 2-133 – Cell fragment indication

Value	Description
'11'	A single cell carrying a complete metadata Access Unit.
'10'	The first cell from a series of cells with data from one metadata Access Unit.
'01'	The last cell from a series of cells with data from one metadata Access Unit.
'00'	A cell from a series of cells with data from one metadata Access Unit, but neither the first nor the last one.

random_access_indicator: This 1-bit field, when coded with the value '1', indicates that the metadata carried in this metadata_AU_cell represents an entry point to the metadata service where decoding is possible without information from previous metadata_AU_cells. The meaning of a random access point is defined by the format of the metadata.

decoder_config_flag: This 1-bit field signals the presence of decoder configuration information in the carried metadata Access Unit. Note that this does not preclude the presence of metadata in the Access Unit next to decoder configuration data.

AU_cell_data_length: This 16-bit field specifies the number of AU_cell_data_bytes immediately following.

AU_cell_data_byte: This 8-bit field contains contiguous bytes from a metadata Access Unit.

2.12.5 Use of the DSM-CC synchronized download protocol to transport metadata

For synchronized transport, in addition to PES packets, the DSM-CC synchronized download protocol can be used. When using synchronized DSM-CC download sections to transport the metadata, the Metadata Access Unit Wrapper defined in 2.12.4.2 shall be used as the tool to encapsulate metadata Access Units. This allows random access indication, whose meaning depends on the format of the metadata, and a cell sequence counter to identify loss of metadata_AU_cells. In each DSM-CC synchronized download section that carries metadata, the first byte of the payload shall be the first byte of a Metadata_AU_cell. For each metadata Access Unit contained in the same DSM-CC synchronized download section, the PTS in the section header applies. The PTS signals the time at which the metadata

Access Units are decoded instantaneously and removed from buffer B_n in the STD. Note that the relationship between a decoded metadata Access Unit and audiovisual content is beyond the scope of this Specification. A specific stream_type value (as detailed in Table 2-34) is assigned to signal carriage of metadata in DSM-CC synchronized download sections.

2.12.6 Use of metadata sections to transport metadata

If asynchronous transport of metadata Access Units without a carousel delivery mechanism is needed, metadata sections can be utilized. The syntax and semantics of metadata sections are defined in this subclause. Each metadata section shall carry either one complete metadata Access Unit or a single part of one metadata Access Unit, as signalled by the section_fragment_indication field (see Table 2-134).

For transport in metadata sections, the metadata Access Units are structured in one or more Metadata Tables. Each Metadata Table contains one or more complete metadata Access Units from one or more metadata services. Conceptually, the transport mechanism of Metadata Tables is comparable to the transport mechanism of Program Map Tables and Program Association Tables. Each Metadata Table may be made up of multiple metadata sections. Each Metadata Table may contain metadata from multiple metadata services.

Specific stream_type and table_id values are assigned to metadata sections. Metadata decoder configuration data can also be carried in sections, signalled by a metadata description value, as assigned by the metadata decoder configuration descriptor.

Table 2-134 – Section syntax for transport of metadata

Syntax	No. of bits	Mnemonic
Metadata_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
private_indicator	**1**	**bslbf**
random_access_indicator	**1**	**bslbf**
decoder_config_flag	**1**	**bslbf**
metadata_section_length	**12**	**uimsbf**
metadata_service_id	**8**	**uimsbf**
reserved	**8**	**bslbf**
section_fragment_indication	**2**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
for (i = 1; i < N; i++){		
metadata_byte	**8**	**bslbf**
}		
CRC_32	**32**	**rpchof**
}		

table_id: The table_id is an 8-bit field that shall be set to '0x06' for each metadata section.

section_syntax_indicator: This 1-bit field shall be set to '1'.

private_indicator: This 1-bit field is not specified by this Specification.

random_access_indicator: This 1-bit field, when coded with the value '1', indicates that the metadata carried in this metadata section represents an access point to the metadata service where decoding is possible without information from previous metadata sections. The meaning of a random access point is defined by the format of the metadata.

decoder_config_flag: This 1-bit field, when coded with the value '1', indicates that decoder configuration information is present in the metadata Access Unit carried in this metadata section.

metadata_section_length: This 12-bit field shall specify the number of remaining bytes in the section immediately following the metadata_section_length field, and including the CRC. The value of this field shall not exceed 4093 (0xFFD).

metadata_service_id: This 8-bit field identifies the metadata service associated with the metadata Access Unit carried in this metadata section. Each Metadata Table may contain metadata from multiple metadata services.

section_fragment_indication: This 2-bit field conveys information on the fragmentation of the metadata Access unit carried in this metadata section, corresponding to Table 2-135.

Table 2-135 – Section fragment indication

Value	Description
'11'	A single metadata section carrying a complete metadata Access Unit.
'10'	The first metadata section from a series of metadata sections with data from one metadata Access Unit.
'01'	The last metadata section from a series of metadata sections with data from one metadata Access Unit.
'00'	A metadata section from a series of metadata sections with data from one metadata Access Unit, but neither the first nor the last one.

version_number: This 5-bit field is the version number of the whole Metadata Table. The version number shall be incremented by 1 modulo 32 whenever the information contained within the Metadata Table changes. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable Metadata Table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable Metadata Table.

current_next_indicator: A 1-bit field, which when set to '1' indicates that the Metadata Table sent is currently applicable. When the bit is set to '0', it indicates that the Metadata Table sent is not yet applicable and shall be the next Metadata Table to become valid.

section_number: This 8-bit field gives the number of the metadata section. The section_number of the first section in a Metadata Table shall be 0x00. The section_number shall be incremented by 1 with each additional section in this Metadata Table.

last_section_number: This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the complete Metadata Table of which this section is a part.

metadata_byte: This 8-bit contains contiguous bytes from a metadata Access Unit.

CRC_32: This 32-bit field shall contain the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire metadata_section.

2.12.7 Use of the DSM-CC data carousel to transport metadata

The DSM-CC tools as defined in ISO/IEC 13818-6 for Data Carousels can be used if a carousel delivery mechanism is required without the need to express the hierarchical organization of the metadata structure in the transport mechanism. Information on the carousel in which the metadata is contained, is included in the metadata descriptor defined in 2.6.60 and 2.6.62. A specific stream_type value is assigned to signal carriage of metadata in the DSM-CC data carousel. Note that signalling of metadata services within a DSM-CC data carousel is required, but not defined by this Specification.

2.12.8 Use of the DSM-CC object carousel to transport metadata

If a carousel delivery mechanism is required with the capability to express the hierarchical organization of the metadata structure in the transport, then the DSM-CC tools and file structures as defined in ISO/IEC 13818-6 for User to User Object Carousels can be used. These file structures provide the tools to structure the metadata as deemed appropriate for efficient parsing of the metadata and for expressing the hierarchical organization of the metadata. Information needed to identify the carousel in which the metadata is contained, is included in the metadata descriptor defined in 2.6.60 and 2.6.61. This may be the IOP:IOR() as defined in 11.3.1 and 5.7.2.3 of ISO/IEC 13818-6 DSM-CC. A specific stream_type value is assigned to signal carriage of metadata in the DSM-CC object carousel. Note that signalling of metadata services within a DSM-CC object carousel is required, but not defined by this Specification.

2.12.9 Metadata-related signalling

2.12.9.1 General

Metadata-related signalling covers four distinct areas:

- signalling of metadata services and streams;
- signalling of content for use by a metadata system;
- association of metadata to content; and
- signalling of decoder configuration data.

2.12.9.2 Signalling of metadata services and streams

Carriage of metadata is signalled by a stream_type value in the inclusive range between 0x15 and 0x19, specifying which of the five methods described in 2.12.4 to 2.12.8 is used to transport the metadata, and if appropriate, by a stream_id value of 0xFC indicating a metadata stream.

To uniquely identify a metadata service a metadata_service_id value is assigned to each such service by the transport; the assigned value shall be unique within the transport or program stream carrying the metadata service. If the metadata is carried in PES packets with a stream_id of 0xFC, or in metadata sections, or in ISO/IEC 13818-6 synchronized download sections, the assigned metadata_service_id value is signalled explicitly in the header of the metadata_AU_cell or the metadata section. If a ISO/IEC 13818-6 carousel is used to carry the metadata, then the signalling of metadata services is left to the application. The metadata descriptor specifies the format of the metadata and provides information on the decoder configuration data, and is linked to the metadata service by carrying information on the metadata service it is associated with.

2.12.9.3 Signalling of content for use by a metadata system

In 2.6.56 and 2.6.57, a content labelling descriptor is defined that can be used to assign a metadata application format specific reference, the content_reference_id_record, to audiovisual or any other content carried over an MPEG-2 transport stream or program stream. The content_reference_id_record can be used by the metadata system as a label to refer to such content. The content may represent, for example, a program or a stream or segments thereof. The content labelling descriptor also provides information on the content time base used for time referencing from the metadata, including the constant offset in time between the metadata time base and the applied content time base. The descriptor allows carriage of private data. The metadata_application_format may define constraints on the content_reference_record, such as constraints on the time period during which it is valid.

2.12.9.4 Association of metadata to content

In 2.6.58 and 2.6.59 the metadata pointer descriptor is defined to associate a single metadata service to audiovisual or any other content in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream. The metadata is associated with the content within the context as defined by the location of the descriptor. In a transport stream, the descriptor may be located in the PMT in the descriptor loop for either the program or an elementary stream, but may also be located in tables not defined in this Specification, such as tables describing bouquets of broadcast services.

The metadata pointer descriptor points from the content's context to the metadata service associated with that content. The descriptor provides the value of the metadata_service_id that is assigned to the associated metadata service, as well as one or more locations of the associated metadata. The location may for example be within the same transport stream as the content, or within another transport stream, but also at a non-Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream location such as the Internet.

2.12.9.5 Signalling decoder configuration data

Decoding of metadata may require the availability of metadata decoder configuration data. If needed, decoder configuration data shall be contained in one of the metadata services in the same program in the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream as the metadata service. If decoder configuration data is needed to decode a metadata service, then the metadata descriptor either carries such data or provides the information on retrieval of the decoder configuration data from the same or another metadata service. In a transport stream such other service can be found by searching in the PMT for a metadata_descriptor with the metadata_service_id as specified in the decoder_config_metadata_service_id field (and with the same metadata_format and the same metadata_application format).

2.12.9.6 Overview of metadata signalling

Figure 2-9 provides an example of metadata signalling, in which a single program, the "content program", carries the content (or essence) while the metadata is carried in a separate program, the "metadata program". In this example, the metadata program and the content program exist on the same transport stream.

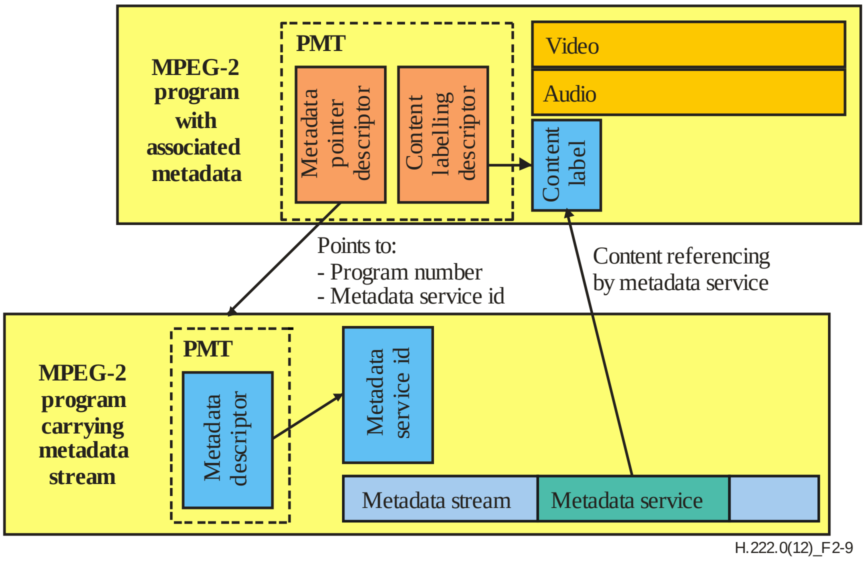


Figure 2-9 – Metadata signalling and referencing

In the content program there are two metadata-related descriptors, the content_labeling descriptor and the metadata_pointer descriptor. The content_labeling descriptor associates a label, illustrated in the diagram by "content label" and encoded in the descriptor in the content_reference_id fields, with the content. The label can then be used by the metadata service to refer to the essence, either in whole, in part, or by a time-described segment. For example, the content_labeling descriptor could provide the label "News of 1/1/02", and the metadata could then refer to a specific story item in the "News of 1/1/02", for example by providing the specific timing of the story item.

The metadata pointer descriptor provides information of where the metadata service can be found for the given content. In this example, the metadata is carried in a separate program, but it would be equally valid to have the metadata carried in the same program as the content, or provided by some means beyond the scope of this Specification, for instance from a URL. This descriptor also provides the metadata service id value that is assigned to the metadata service. This is required since a metadata stream could carry multiple metadata services for many different programs and each program needs to be able to uniquely identify its own metadata service.

In the metadata program, the metadata descriptor signals to which metadata service within a metadata stream it applies. If used, the metadata descriptor provides details of where to find the decoder configuration information.

Upon identifying a metadata pointer descriptor in the PMT by a receiver decoding the content program, the receiver retrieves the metadata descriptor from the metadata program. If needed first the decoder configuration data is retrieved, then the decoder is configured accordingly, after which the metadata service can start being decoded.

2.12.10 STD model for metadata

The STD model specifies normative constraints on Rec. ITU-T H.222.0 | ISO/IEC 13818-1 streams that carry metadata. For decoding of metadata in the STD, the regular T-STD and P-STD models are applicable with buffer B_n, input rate Rx_n of the metadata into B_n and output rate $R_{metadata}$ out of B_n and into $D_{metadata}$, the metadata decoder. See Figure 2-10.

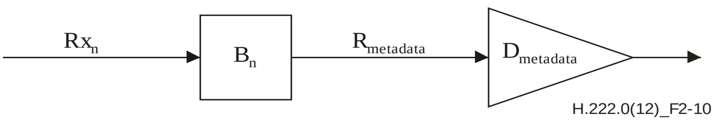


Figure 2-10 – Metadata decoding in the STD

The metadata enters buffer B_n at rate Rx_n. In the P-STD, rate Rx_n equals the rate of the program stream. In the T-STD, rate Rx_n is the rate out of TB_n and equal to the rate defined by the metadata_input_leak_rate field in the metadata STD descriptor. The size BS_n of buffer B_n is equal to the size defined in the metadata_buffer_size field in the metadata STD descriptor. In case of synchronous delivery, metadata decoding is instantaneous and controlled by PTSs. At decode time, that is when the STC equals the PTS, the associated metadata is removed instantaneously from B_n. In case of asynchronous delivery, the metadata is removed from B_n at a rate $R_{metadata}$ equal to the rate defined by the metadata_output_leak_rate field in the metadata STD descriptor. Buffer B_n shall not overflow.

Note that the STD model defines constraints on the delivery of the metadata, without specifying any constraint on the timing used in the metadata.

2.13 Carriage of ISO 15938 data

2.13.1 Introduction

Carriage of metadata over a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream as defined in 2.12 allows for carriage of ISO 15938 data by appropriate coding of the metadata_format field. In this subclause, for the purpose to transport ISO 15938 data, a specific instance is defined. Carriage of ISO 15938 data shall meet each requirement defined in 2.12, but in addition the requirements defined in this subclause shall apply for transport of ISO 15938 data.

2.13.2 ISO 15938 decoder configuration data

Decoding of ISO 15938 data requires the availability of decoder configuration data. Consequently, when ISO 15938 data is carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, then the metadata descriptor shall signal carriage of associated decoder configuration data in the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream by coding a value of the decoder_config_flags of either '001' or '010' or '011' or '100'.

2.14 Carriage of Rec. ITU-T H.264 | ISO/IEC 14496-10 video

2.14.1 Introduction

This Specification defines the carriage of Rec. ITU-T H.264 | ISO/IEC 14496-10 elementary stream within Rec. ITU-T H.222.0 | ISO/IEC 13818-1 systems, both for program and transport streams. Typically, a Rec. ITU-T H.264 | ISO/IEC 14496-10 stream will be an element of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, as defined by the PMT in a transport stream and the PSM in a program stream. The carriage and buffer management of AVC video streams is defined using existing parameters from this Recommendation | International Standard such as PTS and DTS, as well as information present within an AVC video stream.

Carriage of AVC video streams in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream defines accurate mapping between STD parameters and HRD parameters that may be present in an AVC video stream. Requirements are defined for the presence of HRD parameters in the AVC video stream, to ensure that it can be verified whether each STD requirement is met for each AVC video stream carried in a transport stream or a program stream.

NOTE 1 – Though the timing information present in the AVC video stream may not use a 90-kHz clock, the PTS and DTS timestamps need to be expressed in units of 90 kHz.

When a Rec. ITU-T H.264 | ISO/IEC 14496-10 stream is carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the Rec. ITU-T H.264 | ISO/IEC 14496-10 coded data shall be contained in PES packets. The Rec. ITU-T H.264 | ISO/IEC 14496-10 coded data shall comply with the byte stream format defined in Annex B of Rec. ITU-T H.264 | ISO/IEC 14496-10, with the following constraints:

- Each AVC access unit shall contain an access unit delimiter NAL Unit;

  NOTE 2 – Rec. ITU-T H.264 | ISO/IEC 14496-10 requires that an access unit delimiter NAL Unit, if present, is the first NAL Unit within an AVC access unit. Access unit delimiter NAL Units simplify the ability to detect the boundary between pictures; they avoid the need to process the content of slice headers, and they are particularly useful for the Baseline and Extended profiles where slice order can be arbitrary.

- All Sequence and Picture Parameter Sets (SPS and PPS) necessary for decoding the AVC video stream shall be present within that AVC video stream.

  NOTE 3 – Rec. ITU-T H.264 | ISO/IEC 14496-10 also allows delivery of SPS and PPS by external means. This Specification does not provide support for such delivery, and therefore requires SPS and PPS to be carried within the AVC video stream.

- Each AVC video sequence that contains hrd_parameters() with the low_delay_hrd_flag set to '1', shall carry VUI parameters in which the timing_info_present_flag shall be set to '1'.

  NOTE 4 – If the low_delay_hrd_flag is set to '1', then buffer underflow is allowed to occur in the STD model; see 2.14.3 and 2.14.4. Setting the timing_info_present_flag to '1' ensures that the AVC video stream contains sufficient information to determine the DPB output time and the CPB removal time of AVC access units, also in case of underflow.

To provide display specific information such as aspect_ratio, it is strongly recommended that each AVC video stream carries VUI parameters with sufficient information to ensure that the decoded AVC video stream can be displayed correctly by receivers.

When an AVC video stream conforms to one or more profiles defined in Annex G of Rec. ITU-T H.264 | ISO/IEC 14496-10, the following constraints additionally apply:

- The AVC video sub-bitstream of SVC as defined in 2.1.10 shall be an element of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program and the stream_type for this elementary stream shall be equal to 0x1B.
- For each SVC video sub-bitstream as defined in 2.1.121 that is an element of the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, the stream_type for this elementary stream shall be equal to 0x1F.
- All subset Sequence Parameter Sets and Picture Parameter Sets necessary for decoding an SVC video sub-bitstream shall be present within the elementary stream carrying the SVC video sub-bitstream.
- In each elementary stream with stream_type equal to 0x1F, exactly one VDRD_drd_nal_unit as defined in 2.14.3.3 may precede all the NAL units of the same SVC dependency representation.

  NOTE 5 – If any VDRD_drd_nal_unit is included in any SVC dependency representation then the HRD model should include this VDRD_drd_nal_unit in the buffer model as additional non-VCL NAL units. The NAL unit type 24 may be used in a different way by other specifications out of scope of this Specification. When carrying AVC base and SVC enhancement layers in different elementary streams, usage of VDRD is strongly recommended if access units are not aligned with PES packets.
- The TREF field as defined in 2.4.3.7 may be present in the PES headers of elementary streams with stream_type equal to 0x1F. The TREF field shall be set and shall be present in the PES headers as specified in 2.14.3.5 and 2.14.3.6 respectively.

  NOTE 6 – Currently the presence of TREF is only specified for elementary streams with stream_type equal to 0x1F.
- When a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program includes more than one SVC video sub-bitstream, or more than one AVC video sub-bitstream of SVC and at least one SVC video sub-bitstream, a hierarchy descriptor as defined in 2.6.7 shall be used to indicate the dependencies of the related video sub-bitstreams.
- All NAL units of a re-assembled AVC access unit shall be passed to the decoder in the order of NAL units within an access unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10.

  NOTE 7 – If SEI NAL units are present in any SVC dependency representation of an SVC video sub-bitstream, these NAL units may require re-ordering to the order of NAL units within an access unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 before access unit re-assembling.
- The profile and level limitations indicated by profile_idc and level_idc syntax elements in the AVC_video_descriptor, if present, and the Type II HRD parameters in the AVC_timing_and_HRD_descriptor, if present, for an AVC video stream resulting from re-assembling (up to) the video sub-bitstream associated with the descriptors shall include NAL units with nal_unit_type syntax element equal to 14 in the AVC video sub-bitstream of SVC and, if present in the SVC video sub-bitstream, NAL units with nal_unit_type syntax element equal to 24.

When an AVC video stream conforms to one or more profiles defined in Annex H of Rec. ITU-T H.264 | ISO/IEC 14496-10, the following constraints additionally apply:

- The AVC video sub-bitstream of MVC or MVC base view sub-bitstream, as defined in 2.1.9 and 2.1.83, shall be an element of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program and the stream_type for this elementary stream shall be equal to 0x1B.
- For each MVC video sub-bitstream, as defined in 2.1.86, that is an element of the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, the stream_type for this elementary stream shall be equal to 0x20.
- Each MVC video sub-bitstream shall be associated with one or more consecutive view order index values.

  NOTE – According to its definition in 2.1.86, an MVC video sub-bitstream or MVC base view sub-bitstream does not necessarily include view components for all view_id values included in one MVC view_id subset if one or more views are not required for decoding the transmitted views. As an example, consider a MVC bitstream having 4 views V1, V2, V3, and V4 in ascending order of view order index, where view V1 is the base view, view V2 is depending directly on V1, V3 is depending directly on V1 and V2, and V4 is depending directly on V2. Using such encoded views, two MVC sub-bitstreams M1 and M2 may be created as follows: M1 is associated with the output views V1 and V2, and M2 with the output view V4. In this example, it is possible that only M1 and M2 are transmitted to a receiver, thus sub-bitstream for V3 is not required to be transmitted since a combination of both sub-bitstreams M1 and M2 refers to the set of views V1, V2 and V4, and can be decoded without the presence of V3.
- Each view order index value shall be associated with exactly one MVC view_id subset.

  NOTE 8 – This restriction greatly simplifies the re-assembly of any decodable sub-bitstream.
- When a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program includes more than one MVC video sub-bitstream or more than one AVC video sub-bitstream of MVC and at least one MVC video sub-bitstream, one or more hierarchy descriptors as defined in 2.6.6 and 2.6.7 shall be used to indicate

the dependencies of the related video sub-bitstreams. If more than one hierarchy descriptor is present for one elementary stream, the value of the syntax element hierarchy_layer_index shall be the same within the same elementary stream. The syntax element hierarchy_type shall be set to the value 9 or 15.

NOTE – Provided an MVC video sub-bitstream B depends on video sub-bitstream A and this dependency is indicated using a hierarchy descriptor, further an MVC video sub-bitstream C depends on B and this dependency is also indicated using a second hierarchy descriptor, then this implicitly indicates a dependency of C on A and no third hierarchy descriptor is needed.

- The subset sequence parameter sets and picture parameter sets necessary for decoding an MVC video sub-bitstream shall be present within the elementary stream carrying the MVC video sub-bitstream.
- In each elementary stream with stream type equal to 0x20 exactly one VDRD_NAL_unit, as defined in 2.14.3.3, may precede all the NAL units of the same MVC view-component subset.

  NOTE 9 – If any VDRD_nal_unit is included in any MVC view component subset, then the HRD model should include this VDRD_nal_unit in the buffer model as additional non-VCL NAL units. The NAL unit type 24 may be used in a different way by other specifications out of scope of this Specification.

- All NAL units of a re-assembled AVC access unit shall be passed to the decoder in the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10.

  NOTE 10 – If SEI NAL units are present in any MVC view-component subset of an MVC video sub-bitstream, these NAL units may require re-ordering to the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 before access unit re-assembling.

- The profile_idc and level_idc indication in the AVC_video_descriptor, if present, and the Type II HRD parameters in the AVC_timing_and_HRD_descriptor, if present, for an AVC video stream resulting from re-assembling (up to) the MVC video sub-bitstream associated with the descriptors shall include NAL units with nal_unit_type syntax element equal to 14, if present, in the AVC video sub-bitstream of MVC or MVC base view sub-bitstream and, if present, in the MVC video sub-bitstream, NAL units with nal_unit_type syntax element equal to 20 and 24.

When an AVC video stream conforms to one or more profiles defined in Annex I of Rec. ITU-T H.264 | ISO/IEC 14496-10, the following constraints additionally apply:

- The AVC video sub-bitstream of MVCD or MVCD base view sub-bitstream, as defined in 2.1.10 and 2.1.88, shall be an element of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program and the stream_type for this elementary stream shall be equal to 0x1B.
- For each MVCD video sub-bitstream, as defined in 2.1.90, that is an element of the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, the stream_type for this elementary stream shall be equal to 0x26.
- Each MVCD video sub-bitstream shall be associated with one or more consecutive view order index values.
- Each view order index value shall be associated to exactly one MVCD view_id subset.

NOTE 11 – This restriction greatly simplifies the re-assembly of any decodable sub-bitstream.

- The subset sequence parameter sets and picture parameter sets necessary for decoding an MVCD video sub-bitstream shall be present within the elementary stream carrying the MVCD video sub-bitstream.
- In each elementary stream with stream type equal to 0x26 exactly one VDRD_ nal_unit, as defined in 2.14.3.3, may precede all the NAL units of the same MVCD view-component subset.

  NOTE 12 – If any VDRD_ nal_unit is included in any MVCD view component subset, then the HRD model should include this VDRD_nal_unit in the buffer model as additional non-VCL NAL units. The NAL unit type 24 may be used in a different way by other specifications out of scope of this Specification.

- All NAL units of a re-assembled AVC access unit shall be passed to the decoder in the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE 13 – If SEI NAL units are present in any MVCD view-component subset of an MVCD video sub-bitstream, these NAL units may require re-ordering to the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 before access unit re-assembling.

2.14.2 Carriage in PES packets

Rec. ITU-T H.264 | ISO/IEC 14496-10 Video is carried in PES packets as PES_packet_data_bytes, using one of the 16 stream_id values assigned to video, while signalling the Rec. ITU-T H.264 | ISO/IEC 14496-10 Video stream by means of the assigned stream-type value in the PMT or PSM (see Table 2-34). The highest level that may occur in an AVC video stream as well as a profile that the entire stream conforms to should be signalled using the AVC video descriptor. If an AVC video descriptor is associated with an AVC video stream, then this descriptor shall be conveyed in the descriptor loop for the respective elementary stream entry in the Program Map Table in case of a transport stream or in the Program Stream Map, when PSM is present, in case of a program stream. This Recommendation | International Standard does not specify presentation of Rec. ITU-T H.264 | ISO/IEC 14496-10 streams in the context of a program.

For PES packetization, no specific data alignment constraints apply. For synchronization and STD management, PTSs and, when appropriate, DTSs are encoded in the header of the PES packet that carries the Rec. ITU-T H.264 | ISO/IEC 14496-10 video elementary stream data. For PTS and DTS encoding, the constraints and semantics apply as defined in 2.4.3.7 and 2.7.

2.14.3 STD extensions

2.14.3.1 T-STD extensions

The T-STD model includes a transport buffer TB_n and a multiplex buffer MB_n prior to buffer EB_n for decoding of each AVC video elementary stream n conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 video elementary stream n. See Figure 2-11.

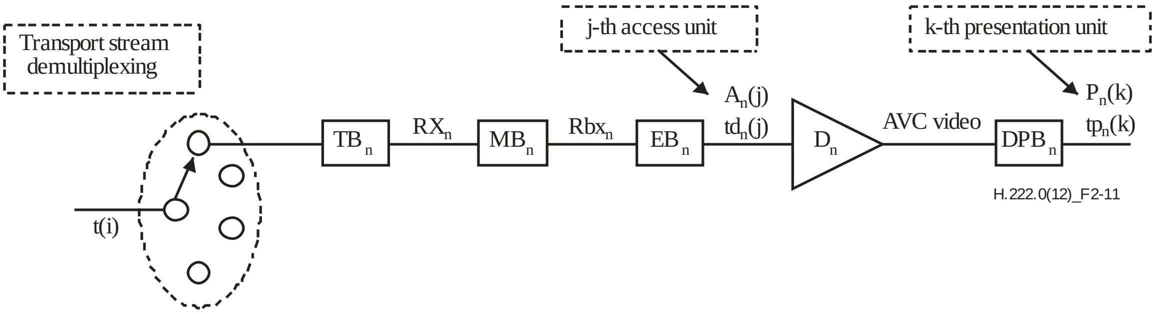


Figure 2-11 – T-STD model extensions for Rec. ITU-T H.264 | ISO/IEC 14496-10 video

DPB_n buffer management

Carriage of an AVC video stream over Rec. ITU-T H.222.0 | ISO/IEC 13818-1 does not impact the size of buffer DPB_n. For decoding of an AVC video stream in the STD the size of DPB_n is as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10. The DPB buffer shall be managed as specified in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10 (C.2 and C.4). A decoded AVC access unit enters DPB_n instantaneously upon decoding of the AVC access unit, hence at the CPB removal time of the AVC access unit. A decoded AVC access unit is presented at the DPB output time. If the AVC video stream provides insufficient information to determine the CPB removal time and the DPB output time of AVC access units, then these time instants shall be determined in the STD model from PTS and DTS timestamps as follows:

1) The CPB removal time of AVC access unit n is the instant in time indicated by DTS(n) where DTS(n) is the DTS value of AVC access unit n.
2) The DPB output time of AVC access unit n is the instant in time indicated by PTS(n) where PTS(n) is the PTS value of AVC access unit n.

NOTE 1 – AVC video sequences in which the low_delay_hrd_flag in hrd parameters() is set to 1 carry sufficient information to determine the DPB output time and the CPB removal time of each AVC access unit. Hence for AVC access units for which STD underflow may occur, the CPB removal time and the DPB output time are defined by HRD parameters, and not by DTS and PTS timestamps.

TB_n, MB_n and EB_n buffer management

The input to buffer TB_n and its size TBS_n are specified in 2.4.2.4. For buffers MB_n and EB_n, and for the rate Rx_n between TB_n and MB_n and the rate Rbx_n between MB_n and EB_n the following constraints apply for carriage of a Rec. ITU-T H.264 | ISO/IEC 14496-10 stream:

Size EBS_n of buffer EB_n:

EBS_n = cpb_size

Where cpb_size is the size CpbSize[cpb_cnt_minus1] of the CPB for the byte stream format signalled in the NAL hrd_parameters() carried in VUI parameters in the AVC video stream. If NAL hrd_parameters() are not present in the AVC video stream, then the cpb_size shall be the size defined as 1200 × MaxCPB in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 for the level of the AVC video stream.

Size MBS_n of Buffer MB_n:

$$MBS_n = BS_{mux} + BS_{oh} + 1200 \times MaxCPB[level] - cpb\_size$$

where BS_{oh}, packet overhead buffering, is defined as:

$$BS_{oh} = (1/750) \text{ seconds} \times \max\{1200 \times MaxBR[level], 2\,000\,000 \text{ bit/s}\}$$

and BS_{mux}, additional multiplex buffering, is defined as:

$$BS_{mux} = 0.004 \text{ seconds} \times \max\{1200 \times MaxBR[level], 2\,000\,000 \text{ bit/s}\}$$

where MaxCPB[level] and MaxBR[level] are defined for the byte stream format in Table A.1 (Level Limits) in Rec. ITU-T H.264 | ISO/IEC 14496-10 for the level of the AVC video stream, and

where cpb_size is the size CpbSize[cpb_cnt_minus1] of the CPB for the byte stream format signalled in the NAL hrd_parameters() carried in VUI parameters in the AVC video stream. If NAL hrd_parameters() are not present in the AVC video stream, then the cpb_size shall be the size 1200 × MaxCPB defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 for the level of the AVC video stream.

Rate Rx_n:

when there is no data in TB_n then Rx_n is equal to zero.

Otherwise: Rx_n = bit_rate

where bit_rate is 1.2 × BitRate[SchedSelIdx] of data flow into the CPB for the byte stream format and BitRate[SchedSelIdx] is as defined in Annex E of Rec. ITU-T H.264 | ISO/IEC 14496-10.

NOTE 2 – Annex E specifies the values for BitRate[SchedSelIdx] when NAL_hrd_parameters() is present in the VUI parameters of the AVC video stream and default values for BitRate[SchedSelIdx] based on profile and level when NAL_hrd_parameters() is not present.

Transfer between MB_n and EB_n

If the AVC_timing_and_HRD_descriptor is present with the hrd_management_valid_flag set to '1', then the transfer of data from MB_n to EB_n shall follow the HRD defined scheme for data arrival in the CPB as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

Otherwise, the leak method shall be used to transfer data from MB_n to EB_n as follows:

Rate Rbx_n:

$$Rbx_n = 1200 \times MaxBR[level]$$

where MaxBR[level] is defined for the byte stream format in Table A.1 (Level Limits) in Rec. ITU-T H.264 | ISO/IEC 14496-10 for each level.

If there is PES packet payload data in MB_n, and buffer EB_n is not full, the PES packet payload is transferred from MB_n to EB_n at a rate equal to Rbx_n. If EB_n is full, data are not removed from MB_n. When a byte of data is transferred from MB_n to EB_n, all PES packet header bytes that are in MB_n and precede that byte, are instantaneously removed and discarded. When there is no PES packet payload data present in MB_n, no data is removed from MB_n. All data that enters MB_n leaves it. All PES packet payload data bytes enter EB_n instantaneously upon leaving MB_n.

Removal of AVC access units from EB_n

Each AVC access unit $A_n(j)$ that is present in EB_n is removed instantaneously at time $td_n(j)$. The decoding time $td_n(j)$ is specified by the DTS or from the CPB removal time, as derived from information in the AVC video stream.

STD delay

The total delay of any Rec. ITU-T H.264 | ISO/IEC 14496-10 data other than AVC still picture data through the System Target Decoders buffers TB_n, MB_n, and EB_n shall be constrained by $td_n(j) - t(i) \leq 10$ seconds for all j, and all bytes i in AVC access unit $A_n(j)$.

The delay of any AVC still picture data through the System Target Decoders buffers TB_n, MB_n, and EB_n shall be constrained by $td_n(j) - t(i) \leq 60$ seconds for all j, and all bytes i in AVC access unit $A_n(j)$.

Buffer management conditions

Transport streams shall be constructed so that the following conditions for buffer management are satisfied:

- TB_n shall not overflow and shall be empty at least once every second.

- MB_n, EB_n, and DPB_n shall not overflow.
- EB_n shall not underflow, except when VUI parameters are present for the AVC video sequence with the low_delay_hrd_flag set to '1'. Underflow of EB_n occurs for AVC access unit $A_n(j)$ when one or more bytes of $A_n(j)$ are not present in EB_n at the decoding time $td_n(j)$.

NOTE 3 – An AVC video stream may carry information to determine compliance of the AVC video stream to the HRD, as specified in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10. The presence of this information can be signalled in a transport stream using the AVC timing and HRD descriptor with the hrd_management_valid_flag set to '1'. Irrespective of the presence of this information, compliance of an AVC video stream to the T-STD ensures that HRD buffer management requirements for CPB_n are met when each byte in the AVC video stream is delivered to and removed from CPB_n in the HRD at exactly the same instant in time at which the byte is delivered to and removed from EB_n in the T-STD.

2.14.3.2 P-STD extensions

The P-STD model for the decoding of an AVC video elementary stream n conforming to one or more profiles defined in Annex A of Rec. ITU-T H.264 | ISO/IEC 14496-10 elementary stream includes a multiplex buffer B_n and a decoder D_n followed by a buffer DPB_n (see Figure 2-12). For each AVC video stream n, the size BS_n of buffer B_n in the P-STD is defined by the P-STD_buffer_size field in the PES packet header.

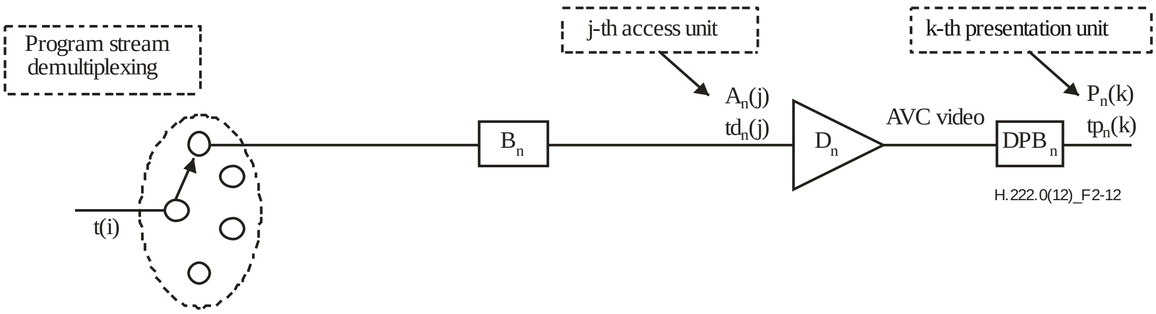


Figure 2-12 – P-STD model extensions for Rec. ITU-T H.264 | ISO/IEC 14496-10 video

DPB_n buffer management

Buffer DPB_n shall be managed in exactly the same way as in the T-STD; see 2.14.3.1.

B_n buffer management

The AVC access unit data enters buffer B_n as specified in 2.5.2.2. At time $td_n(j)$, AVC access unit $A_n(j)$ is decoded and instantaneously removed from B_n. The decoding time $td_n(j)$ is specified by the DTS or by the CPB removal time, derived from information in the AVC video stream. Upon decoding, the AVC access unit instantaneously enters DPB_n or is output without entry into DPB_n, according to the rules specified in Rec. ITU-T H.264 | ISO/IEC 14496-10.

STD delay

The total delay of any Rec. ITU-T H.264 | ISO/IEC 14496-10 data other than AVC still picture data through the System Target Decoders buffer B_n shall be constrained by $td_n(j) – t(i) \leq 10$ seconds for all j, and all bytes i in AVC access unit $A_n(j)$.

The delay of any AVC still picture data through the System Target Decoders buffer B_n shall be constrained by $td_n(j) – t(i) \leq 60$ seconds for all j, and all bytes i in AVC access unit $A_n(j)$.

Buffer management conditions

Program streams shall be constructed so that the following conditions for buffer management are satisfied:

- B_n shall not overflow.
- B_n shall not underflow, except when VUI parameters are present for the AVC video sequence with the low_delay_hrd_flag set to '1' or when trick_mode status is true. Underflow of B_n occurs for AVC access unit $A_n(j)$ when one or more bytes of $A_n(j)$ are not present in B_n at the decoding time $td_n(j)$.

2.14.3.3 View and dependency representation delimiter NAL unit

See Table 2-136.

Table 2-136 – View and dependency representation delimiter NAL unit

Syntax	No. of bits	Mnemonic
VDRD_nal_unit() {		
forbidden_zero_bit	**1**	**bslbf**
nal_ref_idc	**2**	**bslbf**
nal_unit_type	**5**	**bslbf**
}		

2.14.3.4 Semantics of view and dependency representation delimiter NAL unit

forbidden_zero_bit – shall be equal to 0x0

nal_ref_idc – shall be equal to 0x0

nal_unit_type – shall be equal to 0x18

2.14.3.5 T-STD extensions for SVC

The T-STD model described in 2.14.3.1 is applied if the received elementary stream is a video sub-bitstream of stream_type 0x1B, i.e., only the AVC video sub-bitstream of SVC is received and decoded.

When there is a set of received video sub-bitstreams in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, for which dependencies may be signalled in the hierarchy_descriptor as defined in 2.6.6, and when there is at least one of the video sub-bitstreams in the set of received elementary streams having the value of stream_type equal to 0x1F, the T-STD model as described in 2.14.3.1 is extended as illustrated in Figure 2-13 and as specified below.

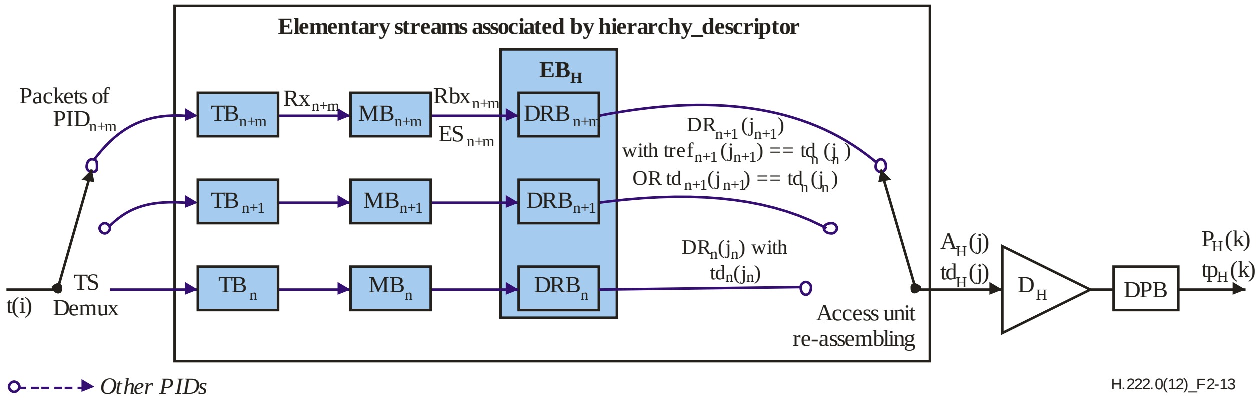


Figure 2-13 – T-STD model extensions for Rec. ITU-T H.264 | ISO/IEC 14496-10 Video with scalable video sub-bitstreams

The following additional notations are used to describe the T-STD extensions and are illustrated in Figure 2-13 above.

- ES_n is the received elementary stream associated with dependency_id value equal to n
- ES_H is the received elementary stream associated with the highest value H of dependency_id present in the set of received elementary streams
- j is an index to the re-assembled access units
- j_n is an index to the SVC dependency representations of the elementary stream associated with dependency_id value equal to n
- $DR_n(j_n)$ is the j_n-th SVC dependency representation of video sub-bitstream associated with dependency_id value equal to n
- $A_H(j)$ is the j-th access unit resulting from re-assembling (up to) the j_H-th SVC dependency representations with dependency_id value equal to H
- $td_n(j_n)$ is the decoding time, measured in seconds, in the system target decoder of the j_n-th SVC dependency representation of the video sub-bitstream associated with dependency_id value equal to n
- $td_H(j)$ is the decoding time, measured in seconds, in the system target decoder of the j-th access unit $A_H(j)$ resulting from re-assembling (up to) the j_H-th SVC dependency representations $DR_H(j_H)$

$tref_n(j_n)$ is equal to the decoding time value $td_{n-1}(j_{n-1})$ associated with the j_{n-1}-th SVC dependency representation $DR_{n-1}(j_{n-1})$, indicated by the TREF field of the PES header of the j_n-th SVC dependency representation $DR_n(j_n)$ of the same access unit

TB_n is the transport buffer for elementary stream associated with dependency_id value equal to n

TBS_n is the size of the transport buffer TB_n, measured in bytes

MB_n is the multiplexing buffer for elementary stream associated with dependency_id value equal to n

MBS_n is the size of the multiplexing buffer MB_n, measured in bytes

DRB_n is the dependency representation buffer for elementary stream ES_n

$DRBS_n$ is the size of dependency representation buffer DRB_n, measured in bytes

EB_H is the elementary stream buffer for all video sub-bitstreams

EBS_H is the size of elementary stream buffer EB_H, measured in bytes

Rx_n is the transfer rate from TB_n to MB_n as specified below

Rbx_n is the transfer rate from MB_n to DRB_n as specified below

Carriage in PES packets

For correct re-assembling of the SVC dependency representations to an AVC access unit, if there is both an SVC dependency representation with any dependency_id equal to n and an SVC dependency representation with dependency_id equal to (n+1) in the same AVC access unit, the following applies:

- a PES packet per SVC dependency representation start shall be present, i.e., at most one SVC dependency representation may commence in the same PES packet;
- the PTS and, if applicable, the DTS value shall be provided in the PES header of each SVC dependency representation;
- if the DTS value of the SVC dependency representation with dependency_id equal to n is different from the DTS value of the SVC dependency representation with dependency_id equal to (n+1) of the same access unit, the TREF field as defined in 2.4.3.7 shall be present in the PES header extension of the SVC dependency representation with dependency_id equal to (n+1) and the TREF field value shall be equal to the DTS value of the SVC dependency representation with dependency_id equal to n.

DPB buffer management

The DPB buffer management for the re-assembled AVC video stream shall conform to 2.14.3.1 using AVC access unit timing values, as DTS or CPB removal time, and PTS or DPB removal time, associated with the SVC dependency representations of the video sub-bitstream in elementary stream ES_H.

TB_n, MB_n, EB_n buffer management

The following applies:

- There is exactly one transport buffer TB as defined in 2.14.3.1 for each received elementary stream in the set of received video sub-bitstreams contained in elementary streams as shown in Figure 2-13.
- There is exactly one multiplexing buffer MB_0 for the AVC video sub-bitstream of SVC in elementary stream ES_0, where the size of the multiplexing buffer MBS_0 is constrained as follows:

$$MBS_0 = BS_{mux,0} + BS_{oh,0} + 1200 \times MaxCPB[level]_0 - cpb\_size_0$$

  where $BS_{mux,0}$, $BS_{oh,0}$ are defined in 2.14.3.1 for the AVC video sub-bitstream of SVC in elementary stream ES_0;

  where $MaxCPB[level]_0$ and cpb_size_0 for the elementary stream ES_0 are defined as in 2.14.3.1.

  NOTE 1 – If HRD parameters are present in at least one of the video sub-bitstreams, those parameters have to be carefully handled in order to not unnecessarily increase the multiplexing buffers allocation.

- There is exactly one multiplexing buffer MB_n for each received elementary stream associated with dependency_id value not equal to 0, where the size of each multiplexing buffer MBS_n is constrained as follows:

$$MBS_n = BS_{mux,n} + BS_{oh,n}$$

  where $BS_{mux,n}$, $BS_{oh,n}$ are defined in 2.14.3.1 for the AVC video stream resulting from re-assembling (up to) the SVC video sub-bitstream in elementary stream ES_n.

- There is exactly one elementary stream buffer EB_H for all the elementary streams in the set of received elementary streams as shown in Figure 2-13, of which the size EBS_H has the following value:

$$EBS_H = cpb\_size_H$$

where cpb_size_H is the cpb_size for the SVC video sub-bitstream in elementary stream ES_H as defined in 2.14.3.1 for the re-assembled AVC video stream.

- There is exactly one dependency representation buffer DRB_n for each elementary stream in the set of received elementary streams as shown in Figure 2-13, where each dependency representation buffer DRB_n in the set of received elementary streams is allocated within EB_H. Even though the size $DRBS_n$ of individual DRB_n is not constrained, the sum of the sizes $DRBS_n$ is constrained as follows:

$$EBS_H = \sum_n (DRBS_n)$$

- Transfer from TB_n to MB_n is applied as follows:

When there is no data in TB_n then Rx_n is equal to zero. Otherwise:

$$Rx_n = bit\_rate$$

where bit_rate is 1.2 x BitRate[SchedSelIdx] of data flow into the CPB for the byte stream format and BitRate[SchedSelIdx] is as defined in Annex E of Rec. ITU-T H.264 | ISO/IEC 14496-10 when NAL hrd_parameters() is present in the VUI parameters of the SVC video sub-bitstream.

NOTE 2 – Annex E also specifies default values for BitRate[SchedSelIdx] based on profile and level when NAL HRD parameters are not present in the VUI. The SVC video sub-bitstream level is determined by the level of AVC video stream resulting from re-assembling (up to) the associated video sub-bitstream n in elementary stream ES_n.

- Transfer from MB_n to DRB_n is applied as follows:

If the AVC_timing_and_HRD_descriptor is present with the hrd_management_valid_flag set to '1' for elementary stream ES_H, then the transfer of data from MB_n to DRB_n shall follow the HRD defined scheme for data arrival in the CPB of elementary stream ES_H as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

Otherwise, the leak method shall be used to transfer data from MB_n to DRB_n as follows:

Rate Rbx_n:

$$Rbx_n = 1200 \times MaxBR[level]_n$$

where $MaxBR[level]_n$ is defined for the byte stream format in Table A.1 (Level Limits) in Rec. ITU-T H.264 | ISO/IEC 14496-10 for the level of the AVC video stream resulting from re-assembling (up to) the associated video sub-bitstream n in elementary stream ES_n. If there is PES packet payload data in MB_n, and buffer EB_H is not full, the PES packet payload is transferred from MB_n to DRB_n at a rate equal to Rbx_n. If EB_H is full, data are not removed from MB_n. When a byte of data is transferred from MB_n to DRB_n, all PES packet header bytes that are in MB_n and precede that byte are instantaneously removed and discarded. When there is no PES packet payload data present in MB_n, no data is removed from MB_n. All data that enters MB_n leaves it. All PES packet payload data bytes enter DRB_n instantaneously upon leaving MB_n.

Access unit re-assembling and EB removal

The following specifies the access unit re-assembling that results in AVC access unit $A_H(j)$:

i) Collect all SVC dependency representations $DR_n(j_n)$ starting with the highest value of dependency_id n, equal to H, to the lowest value of dependency_id n, equal to m, present in access unit $A_H(j)$ following the rule below:

   - For each two corresponding $DR_{y+1}(j_{y+1})$ and $DR_y(j_y)$ of the SVC dependency representations collected for access unit $A_H(j)$, if TREF field is present for $DR_{y+1}(j_{y+1})$, the TREF value of $DR_{y+1}(j_{y+1})$ $tref_{y+1}(j_{y+1})$ shall be equal to the DTS value $td_y(j_y)$ of $DR_y(j_y)$, otherwise the DTS value of $td_{y+1}(j_{y+1})$ of $DR_{y+1}(j_{y+1})$ shall be equal to DTS value $td_y(j_y)$ of $DR_y(j_y)$.

ii) Assemble the SVC dependency representations in consecutive order of dependency_id n starting from 'm' to 'H' for the j-th access unit $A_H(j)$. If SEI NAL units are present in any SVC dependency representation with dependency_id not equal to 0, these NAL units shall be re-ordered to the order of

NAL units within an access unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 before access unit re-assembling.

NOTE 2 – The number of SVC dependency representations for each access unit $A_H(j)$ may vary depending on value of j. If m represents the lowest value of dependency_id for access unit AH(j), then the collected and assembled SVC dependency representations include $DR_m(j_m)$, $DR_{m+1}(j_{m+1})$, …, $DR_H(j_H)$.

The following specifies the removal of access unit $A_H(j)$ from buffer EB_H:

At time $td_H(j)$ the AVC access unit $A_H(j)$ shall be re-assembled and available for removal from buffer EB_H. The decoding time $td_H(j)$ is specified by the DTS or by the CPB removal time that is associated with the SVC dependency representations in elementary stream ES_H, as derived from information in the re-assembled AVC video stream.

STD delay

The STD delay for re-assembled AVC access units shall follow the constraints specified in 2.14.3.1.

Buffer management conditions

Transport streams shall be constructed so that the following conditions for buffer management are satisfied:

- Each TB_n shall not overflow and shall be empty at least once every second.
- Each MB_n, EB_H, and DPB shall not overflow.
- EB_H shall not underflow, except when VUI parameters are present for the AVC video sequence of the re-assembled AVC video stream with the low_delay_hrd_flag set to '1'. Underflow of EB_H occurs for AVC access unit $A_H(j)$ when one or more bytes of $A_H(j)$ are not present in EB_H at the decoding time $td_H(j)$.

2.14.3.6 P-STD extensions for SVC

The P-STD model described in 2.14.3.2 is applied if the decoded elementary stream is a video sub-bitstream of stream_type 0x1B, i.e., only the AVC video sub-bitstream of SVC is decoded.

When there is a set of decoded video sub-bitstreams in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, for which dependencies may be signalled in the hierarchy_descriptor as defined in 2.6.6, and when there is at least one of the video sub-bitstreams in the set of decoded elementary streams having the value of stream_type equal to 0x1F, the P-STD model as described in 2.14.3.2 is extended as illustrated in Figure 2-14 and as specified below.

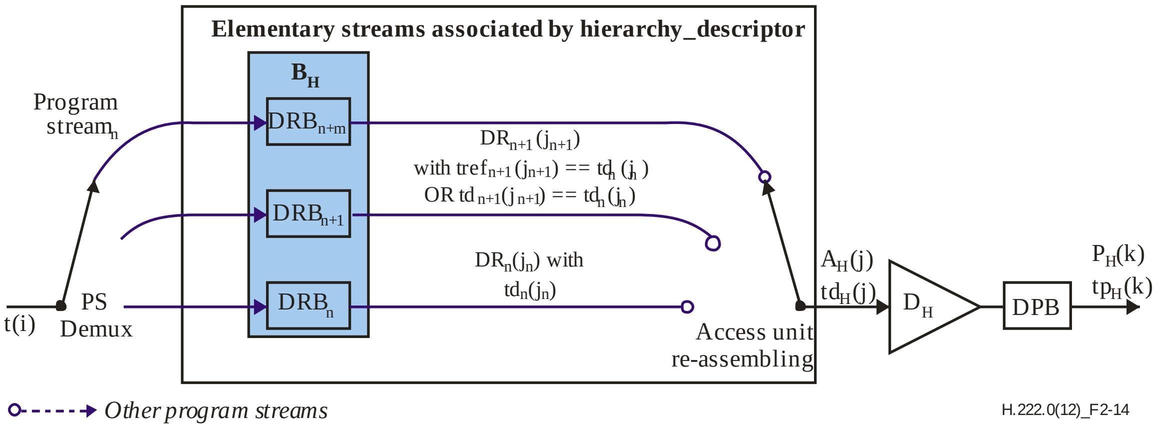


Figure 2-14 – P-STD model extensions for Rec. ITU-T H.264 | ISO/IEC 14496-10 Video with scalable video sub-bitstreams

The following additional notations are used to describe the P-STD extensions and are illustrated in Figure 2-14 above.

ES_n is the decoded elementary stream associated with dependency_id value equal to n

ES_H is the decoded elementary stream associated with the highest value H of dependency_id in the set of decoded elementary streams

j is an index to the re-assembled access units

j_n is an index to the SVC dependency representations of the elementary stream associated with dependency_id value equal to n

$DR_n(j_n)$ is the j_n-th SVC dependency representation of video sub-bitstream associated with dependency_id value equal to n

$A_H(j)$ is the j-th access unit resulting from re-assembling (up to) the j_H-th SVC dependency representations with dependency_id value equal to H

$td_n(j_n)$ is the decoding time, measured in seconds, in the system target decoder of the j_n-th SVC dependency representation of the video sub-bitstream associated with dependency_id value equal to n

$td_H(j)$ is the decoding time, measured in seconds, in the system target decoder of the j-th access unit $A_H(j)$ resulting from re-assembling (up to) the j_H-th SVC dependency representations $DR_H(j_H)$

$tref_n(j_n)$ is equal to the decoding time value $td_{n-1}(j_{n-1})$ associated with the j_{n-1}-th SVC dependency representation $DR_{n-1}(j_{n-1})$, indicated by the TREF field of the PES header of the j_n-th SVC dependency representation $DR_n(j_n)$ of the same access unit

B_H is the input buffer for all decoded video sub-bitstreams

BS_H is the size of the input buffer B_H, measured in bytes

DRB_n is the dependency representation buffer for elementary stream ES_n

$DRBS_n$ is the size of dependency representation buffer DRB_n, measured in bytes

Carriage in PES packets

For correct re-assembling of the SVC dependency representations to an AVC access unit, if there is both an SVC dependency representation with any dependency_id equal to n and an SVC dependency representation with dependency_id equal to (n+1) in the same AVC access unit, the following applies:

- a PES packet per SVC dependency representation start shall be present, i.e., at most one SVC dependency representation may commence in the same PES packet;
- the PTS and, if applicable, the DTS value shall be provided in the PES header of each SVC dependency representation;
- if the DTS value of the SVC dependency representation with dependency_id equal to n is different from the DTS value of the SVC dependency representation with dependency_id equal to (n+1) of the same access unit, the TREF field as defined in 2.4.3.7 shall be present in the PES header extension of the SVC dependency representation with dependency_id equal to (n+1) and the TREF value shall be equal to the DTS value of the SVC dependency representation with dependency_id equal to n.

DPB buffer management

The DPB buffer management for the re-assembled AVC video stream shall conform to 2.14.3.1 using AVC access unit timing values, as DTS or CPB removal time, and PTS or DPB removal time, associated with the SVC dependency representations of the video sub-bitstream in elementary stream ES_H.

B_n buffer management

The following applies:

- There is exactly one elementary stream buffer B_H for all the elementary streams in the set of decoded elementary streams as shown in Figure 2-14, where the size of BS_H is defined by the P-STD_buffer_size field in the PES packet header of elementary stream ES_H.
- There is exactly one dependency representation buffer DRB_n for each elementary stream in the set of decoded elementary streams as shown in Figure 2-14, where each dependency representation buffer DRB_n in the set of decoded elementary streams is allocated within BS_H. Even though the size $DRBS_n$ of individual DRB_n is not constrained, the sum of the sizes $DRBS_n$ is constrained as follows:

$$BS_H = \sum_n (DRBS_n)$$

where BS_H is the size of the input buffer for the SVC video sub-bitstream in elementary stream ES_H as defined in 2.14.3.2 for the re-assembled AVC video stream.

Access unit re-assembling and B removal

The following specifies the access unit re-assembling that results in AVC access unit $A_H(j)$:

i) Collect all SVC dependency representations $DR_n(j_n)$ starting with the highest value of dependency_id n, equal to H, to the lowest value of dependency_id n, equal to m, present in access unit $A_H(j)$ following the rule below:

- For each two corresponding $DR_{y+1}(j_{y+1})$ and $DR_y(j_y)$ of the SVC dependency representations collected for access unit $A_H(j)$, if TREF field is present for $DR_{y+1}(j_{y+1})$, the TREF value of $DR_{y+1}(j_{y+1})$ $tref_{y+1}(j_{y+1})$ shall be equal to the DTS value $td_y(j_y)$ of $DR_y(j_y)$, otherwise the DTS value of $td_{y+1}(j_{y+1})$ of $DR_{y+1}(j_{y+1})$ shall be equal to DTS value $td_y(j_y)$ of $DR_y(j_y)$.

ii) Assemble the SVC dependency representations in consecutive order of dependency_id n starting from 'm' to 'H' for the j-th access unit $A_H(j)$. If SEI NAL units are present in any SVC dependency representation with dependency_id not equal to 0, these NAL units shall be re-ordered to the order of NAL units within an access unit as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10 before access unit re-assembling.

NOTE – The number of SVC dependency representations for each access unit $A_H(j)$ may vary depending on value of j. If m represents the lowest value of dependency_id for access unit $A_H(j)$, then the collected and assembled SVC dependency representations include $DR_m(j_m)$, $DR_{m+1}(j_{m+1})$, …, $DR_H(j_H)$.

The following specifies the removal of access unit $A_H(j)$ from buffer B_H:

At time $td_H(j_H)$ the AVC access unit $A_H(j_H)$ shall be re-assembled and available for removal from buffer B_H. The decoding time $td_H(j)$ is specified by the DTS or by the CPB removal time that is associated with the SVC dependency representations in elementary stream ES_H, as derived from information in the re-assembled AVC video stream.

STD delay

The STD delay for the re-assembled AVC access units shall follow the constraints specified in 2.14.3.2.

Buffer management conditions

Program streams shall be constructed so that the following conditions for buffer management are satisfied:

- B_H shall not overflow.
- B_H shall not underflow, except when VUI parameters are present for the AVC video sequence of the re-assembled AVC video stream with the low_delay_hrd_flag set to '1' or when trick_mode status is true. Underflow of B_H occurs for AVC access unit $A_H(j)$ when one or more bytes of $A_H(j)$ are not present in B_H at the decoding time $td_H(j)$.

2.14.3.7 T-STD extensions for MVC and MVCD

The T-STD model described in 2.14.3.1 is applied if the received elementary stream is a video sub-bitstream of stream_type 0x1B, i.e., only the AVC video sub-bitstream of MVC or MVC base view sub-bitstream is received and decoded.

When there is a set of received video sub-bitstreams and MVC video sub-bitstreams in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, of which dependencies may be signalled in the hierarchy descriptor, as defined in 2.6.7, and when there is at least one of the MVC video sub-bitstreams in the set of received elementary streams having the value of stream_type equal to 0x20, the T-STD model as described in 2.14.3.1 is extended as illustrated in Figure 2-15 and as specified below.

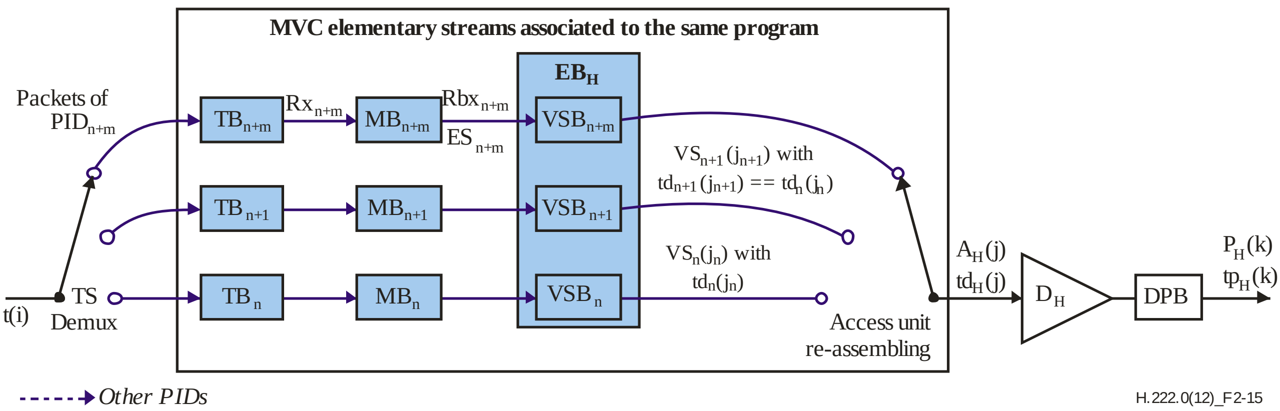


Figure 2-15 – T-STD model extensions for Rec. ITU-T H.264 | ISO/IEC 14496-10 Video with MVC video sub-bitstreams

The following additional notations are used to describe the T-STD extensions and are illustrated in Figure 2-15 above.

ES_n is the received elementary stream associated with the n-th MVC video sub-bitstream, where n is the index to the MVC view_id subsets starting with value 0 for the MVC view_id subset containing the base view and ordered according to the minimum view order index contained in each MVC video sub-bitstream

ES_H is the received elementary stream associated with the H-th MVC video sub-bitstream which includes the view components with the highest view order index present in all MVC video sub-bitstreams of received elementary streams

j is an index to the re-assembled access units

j_n is an index to the MVC view-component subsets of the elementary stream ES_n associated with the n-th MVC video sub-bitstream

$VS_n(j_n)$ is the j_n-th MVC view-component subset of the MVC video sub-bitstream associated with ES_n

$A_H(j)$ is the j-th access unit resulting from re-assembling (up to) the H-th MVC view-component subset associated with ES_H

$td_n(j_n)$ is the decoding time, measured in seconds, in the system target decoder of the MVC view-component subset $VS_n(j_n)$

$td_H(j)$ is the decoding time, measured in seconds, in the system target decoder of the j-th access unit $A_H(j)$ resulting from re-assembling (up to) the MVC view-component subset $VS_H(j_H)$

TB_n is the transport buffer for elementary stream ES_n

TBS_n is the size of the transport buffer TB_n, measured in bytes

MB_n is the multiplexing buffer for elementary stream ES_n

MBS_n is the size of the multiplexing buffer MB_n, measured in bytes

VSB_n is the view component subset buffer for elementary stream ES_n

$VSBS_n$ is the size of view component subset buffer VSB_n, measured in bytes

EB_H is the elementary stream buffer for the AVC video sub-bitstream of MVC and all MVC video sub-bitstreams

EBS_H is the size of elementary stream buffer EB_H, measured in bytes

Rx_n transfer rate from TB_n to MB_n, as specified below

Rbx_n transfer rate from MB_n to VSB_n, as specified below

Carriage in PES packets

For correct re-assembling of the MVC view-component subsets to an AVC access unit, the following applies:

- a PES packet per MVC view-component subset start shall be present, i.e., at most one MVC view-component subset may commence in the same PES packet;
- the PTS and, if applicable, the DTS value shall be provided in the PES header of each MVC view-component subset.

DPB buffer management

The DPB buffer management for the re-assembled AVC video stream shall conform to 2.14.3.1 using AVC access unit timing values, as DTS or CPB removal time, and PTS or DPB removal time, associated with the MVC view-component subsets of the MVC video sub-bitstream in elementary stream ES_H.

TB_n, MB_n, EB_n buffer management

The following applies:

- There is exactly one transport buffer TB, as defined in 2.14.3.1, for each received elementary stream in the set of received MVC video sub-bitstreams, including AVC video sub-bitstream of MVC, contained in elementary streams as shown in Figure 2-15.
- There is exactly one multiplexing buffer MB_0 for the AVC video sub-bitstream of MVC in elementary stream ES_0, where the size of the multiplexing buffer MBS_0 is constrained as follows:

$$MBS_0 = BS_{mux,0} + BS_{oh,0} + 1200 \times MaxCPB[level]_0 - cpb\_size_0$$

where $BS_{mux,0}$, $BS_{oh,0}$ are defined in 2.14.3.1 for the AVC video sub-bitstream of MVC in elementary stream ES_0.

where MaxCPB[level]$_0$ and cpb_size$_0$ for the elementary stream ES_0 are defined, as in 2.14.3.1.

NOTE 1 – If HRD parameters are present in at least one of the MVC video sub-bitstreams, those parameters have to be carefully handled in order to not unnecessarily increase the multiplexing buffers allocation.

- There is exactly one multiplexing buffer MB_n for each received elementary stream associated with view order index value not equal to 0, where the size of each multiplexing buffer MBS_n in the set of received elementary streams is constrained as follows:

$$MBS_n = BS_{mux,n} + BS_{oh,n}$$

where $BS_{mux,n}$, $BS_{oh,n}$ are defined in 2.14.3.1 for the AVC video stream resulting from re-assembling (up to) the MVC video sub-bitstream in elementary stream ES_n.

- There is exactly one elementary stream buffer EB_H for all the elementary streams in the set of received elementary streams as shown in Figure 2-15, of which the size EBS_H has the following value:

$$EBS_H = cpb\_size_H$$

where cpb_size$_H$ is the cpb_size for the MVC video sub-bitstream in elementary stream ES_H as defined in 2.14.3.1 for the re-assembled AVC video stream.

- There is exactly one view component subset buffer VSB_n for each elementary stream in the set of received elementary streams as shown in Figure 2-15, where each view component subset buffer VSB_n in the set of received elementary streams is allocated within EB_H. Even though the size $VSBS_n$ of individual VSB_n is not constrained, the sum of the sizes $VSBS_n$ is constrained as follows:

$$EBS_H = \sum_n (VSBS_n)$$

- Transfer from TB_n to MB_n is applied as follows:

Rate Rx_n:

when there is no data in TB_n, then Rx_n is equal to zero.

Otherwise: Rx_n = bit_rate

where bit_rate is 1.2 × BitRate[SchedSelIdx] of data flow into the CPB for the byte stream format and BitRate[SchedSelIdx] is as defined in Annex E of Rec. ITU-T H.264 | ISO/IEC 14496-10 when NAL hrd_parameters() is present in the VUI parameters of the MVC video sub-bitstream.

NOTE 2 – Annex E also specifies default values for BitRate[SchedSelIdx] based on profile and level when NAL HRD parameters are not present in the VUI. The MVC video sub-bitstream level is determined by the level of AVC video stream resulting from re-assembling (up to) the associated MVC video sub-bitstream n in elementary stream ES_n.

- Transfer from MB_n to VSB_n is applied as follows:

If the AVC_timing_and_HRD_descriptor is present with the hrd_management_valid_flag set to '1' for elementary stream ES_H, then the transfer of data from MB_n to VSB_n shall follow the HRD defined scheme for data arrival in the CPB of elementary stream ES_H as defined in Annex C of Rec. ITU-T H.264 | ISO/IEC 14496-10.

Otherwise, the leak method shall be used to transfer data from MB_n to VSB_n as follows:

Rate Rbx_n:

$$Rbx_n = 1200 \times MaxBR[level]_n$$

where MaxBR[level]$_n$ is defined for the byte stream format in Table A.1 (Level limits) in Rec. ITU-T H.264 | ISO/IEC 14496-10 for the level of the AVC video stream resulting from re-assembling (up to) the associated MVC video sub-bitstream n in elementary stream ES_n. If there is PES packet payload data in MB_n, and buffer EB_H is not full, the PES packet payload is transferred from MB_n to VSB_n at a rate equal to Rbx_n. If EB_H is full, data are not removed from MB_n. When a byte of data is transferred from MB_n to VSB_n, all PES packet header bytes that are in MB_n and precede that byte are instantaneously removed and discarded. When there is no PES packet payload data present in MB_n, no data is removed from MB_n. All data that enters MB_n leaves it. All PES packet payload data bytes enter VSB_n instantaneously upon leaving MB_n.

Access unit re-assembling and EB removal

The following specifies the access unit re-assembling that results in AVC access unit $A_H(j)$:

i) Assemble the MVC view-component subsets for the j-th access unit $A_H(j)$ following the rule below:

- For each two corresponding MVC view-component subsets $VS_{y+1}(j_{y+1})$ and $VS_y(j_y)$ collected for access unit $A_H(j)$, where VS_y is associated with a program element identified by the hierarchy_layer_index indicated in the associated hierarchy descriptor, and VS_{y+1} indicates the hierarchy_layer_index of VS_y as the hierarchy_embedded_layer_index in the hierarchy descriptor associated with program element associated with VS_{y+1}, the DTS value of $td_{y+1}(j_{y+1})$ of $VS_{y+1}(j_{y+1})$ shall be equal to DTS value $td_y(j_y)$ of $VS_y(j_y)$.

  NOTE 3 – If no hierarchy descriptor is present, VS_y is associated with the AVC sub-bitstream and VS_{y+1} is associated with the MVC sub-bitstream.

ii) If SEI NAL units are present in any MVC view-component subset with view order index not equal to 0, these NAL units shall be re-ordered to the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10, before access unit re-assembling.

The following specifies the removal of access unit $A_H(j)$ from buffer EB_H:

At the decoding time $td_H(j)$, the AVC access unit $A_H(j)$ shall be re-assembled and available for removal from buffer EB_H. The decoding time $td_H(j)$ is specified by the DTS or by the CPB removal time that is associated with the MVC view-component subsets in elementary stream ES_H, as derived from information in the re-assembled AVC video stream.

STD delay

The STD delay for re-assembled AVC access units shall follow the constraints specified in 2.14.3.1.

Buffer management conditions

Transport streams shall be constructed so that the following conditions for buffer management are satisfied:

- Each TB_n shall not overflow and shall be empty at least once every second.
- Each MB_n, EB_H, and DPB shall not overflow.
- EB_H shall not underflow, except when VUI parameters are present for the AVC video sequence of the re-assembled AVC video stream with the low_delay_hrd_flag set to '1'. Underflow of EB_H occurs for AVC access unit $A_H(j)$ when one or more bytes of $A_H(j)$ are not present in EB_H at the decoding time $td_H(j)$.

Buffer management for MVCD video sub-bitstream follows the same extensions as specified for MVC video in this clause. Carriage in PES packets for MVCD video sub-bitstream is specified below:

Carriage of MVCD sub-bitstream in PES packets

For correct re-assembling of the MVCD view-component subsets to an AVC access unit, the following applies:

- a PES packet per MVCD view-component subset start shall be present, i.e., at most one MVCD view-component subset may commence in the same PES packet;
- the PTS and, if applicable, the DTS value shall be provided in the PES header of each MVCD view-component subset.

2.14.3.8 P-STD extensions for MVC

The P-STD model described in 2.14.3.2 is applied if the decoded elementary stream is a video sub-bitstream of stream_type 0x1B, i.e., only the AVC video sub-bitstream of MVC or MVC base view sub-bitstream is decoded.

When there is a set of decoded MVC video sub-bitstreams in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, of which view order index values may be signalled in the MVC_extension_descriptor, as defined in 2.6.78, and when there is at least one of the MVC video sub-bitstreams in the set of decoded elementary streams having the value of stream_type equal to 0x20, the P-STD model, as described in 2.14.3.2, is extended as illustrated in Figure 2-16 and as specified below.

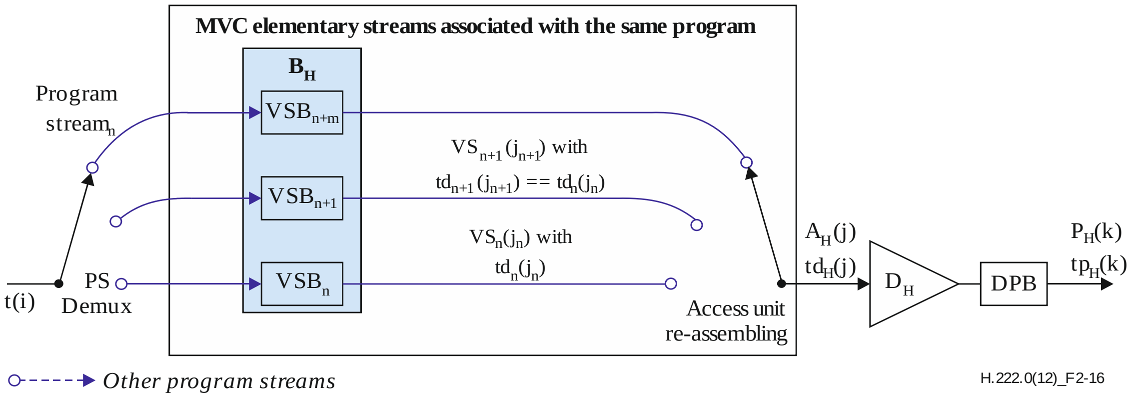


Figure 2-16 – P-STD model extensions for Rec. ITU-T H.264 | ISO/IEC 14496-10 Video with MVC video sub-bitstreams

The following additional notations are used to describe the P-STD extensions and are illustrated in Figure 2-16 above.

ES_n is the received elementary stream associated with the n-th MVC video sub-bitstream, where n is the index to the MVC view_id subsets starting with value 0 for the MVC view_id subset containing the base view and ordered according to the minimum view order index contained in each MVC view_id subset

ES_H is the received elementary stream associated with the H-th MVC video sub-bitstream which includes the view components with the highest view order index present in all MVC view_id subsets of received elementary streams

j is an index to the re-assembled access units

j_n is an index to the MVC view-component subsets of the elementary stream associated with the n-th MVC view_id subset

$VS_n(j_n)$ is the j_n-th MVC view-component subset of the MVC video sub-bitstream associated with ES_n

$A_H(j)$ is the j-th access unit resulting from re-assembling (up to) the H-th MVC view-component subset associated with ES_H

$td_n(j_n)$ is the decoding time, measured in seconds, in the system target decoder of the MVC view-component subset $VS_n(j_n)$

$td_H(j)$ is the decoding time, measured in seconds, in the system target decoder of the j-th access unit $A_H(j)$ resulting from re-assembling (up to) the MVC view-component subset $VS_H(j_H)$

B_H is the input buffer for all decoded MVC video sub-bitstreams

BS_H is the size of the input buffer B_H, measured in bytes

VSB_n is the view component subset buffer for elementary stream ES_n

$VSBS_n$ is the size of view component subset buffer VSB_n, measured in bytes

Carriage in PES packets

For correct re-assembling of the MVC view-component subsets to an AVC access unit, the following applies:

- a PES packet per MVC view-component subset start shall be present, i.e., at most one MVC view-component subset may commence in the same PES packet;
- the PTS and, if applicable, the DTS value shall be provided in the PES header of each MVC view-component subset.

DPB buffer management

The DPB buffer management for the re-assembled AVC video stream shall conform to 2.14.3.1 using AVC access unit timing values, as DTS or CPB removal time, and PTS or DPB removal time, associated with the MVC view-component subsets of the MVC video sub-bitstream in elementary stream ES_H.

B_n buffer management

The following applies:

- There is exactly one elementary stream buffer B_H for all the elementary streams in the set of decoded elementary streams as shown in Figure 2-16, where the size of BS_H is defined by the P-STD_buffer_size field in the PES packet header of elementary stream ES_H.
- There is exactly one view component subset buffer VSB_n for each elementary stream in the set of decoded elementary streams as shown in Figure 2-16, where each view component subset buffer VSB_n in the set of decoded elementary streams is allocated within BS_H. Even though the size $VSBS_n$ of individual VSB_n is not constrained, the sum of the sizes $VSBS_n$ is constrained as follows:

$$BS_H = \sum_n (VSBS_n)$$

where BS_H is the size of the input buffer for the MVC video sub-bitstream in elementary stream ES_H, as defined in 2.14.3.2, for the re-assembled AVC video stream.

Access unit re-assembling and B removal

The following specifies the access unit re-assembling that results in AVC access unit $A_H(j)$:

i) Assemble the MVC view-component subsets for the j-th access unit $A_H(j)$ following the rule below:

- For each two corresponding MVC view-component subsets $VS_{y+1}(j_{y+1})$ and $VS_y(j_y)$ collected for access unit $A_H(j)$, where VS_y is associated with a program element identified by the hierarchy_layer_index indicated in the associated hierarchy descriptor, and VS_{y+1} references VS_y by the hierarchy_embedded_layer_index indicated in the hierarchy descriptor associated with program element associated with VS_{y+1}, the DTS value of $td_{y+1}(j_{y+1})$ of $VS_{y+1}(j_{y+1})$ shall be equal to DTS value $td_y(j_y)$ of $VS_y(j_y)$.

  NOTE – If no hierarchy descriptor is present, VS_y is associated with the AVC sub-bitstream and VS_{y+1} is associated with the MVC sub-bitstream.

ii) If SEI NAL units are present in any MVC view-component subset with view order index not equal to 0, these NAL units shall be re-ordered to the order of NAL units within an access unit, as defined in Rec. ITU-T H.264 | ISO/IEC 14496-10, before access unit re-assembling.

The following specifies the removal of access unit $A_H(j)$ from buffer B_H:

At the decoding time $td_H(j_H)$, the AVC access unit $A_H(j_H)$ shall be re-assembled and available for removal from buffer B_H. The decoding time $td_H(j)$ is specified by the DTS or by the CPB removal time that is associated with the MVC view-component subsets in elementary stream ES_H, as derived from information in the re-assembled AVC video stream.

STD delay

The STD delay for the re-assembled AVC access units shall follow the constraints specified in 2.14.3.2.

Buffer management conditions

Program streams shall be constructed so that the following conditions for buffer management are satisfied:

- B_H shall not overflow.
- B_H shall not underflow, except when VUI parameters are present for the AVC video sequence of the re-assembled AVC video stream with the low_delay_hrd_flag set to '1', or when trick_mode status is true. Underflow of B_H occurs for AVC access unit $A_H(j)$ when one or more bytes of $A_H(j)$ are not present in B_H at the decoding time $td_H(j)$.

2.15 Carriage of ISO/IEC 14496-17 text streams

2.15.1 Introduction

This subclause defines the carriage of ISO/IEC 14496-17 elementary text streams within Rec. ITU-T H.222.0 | ISO/IEC 13818-1 systems, both for program and transport streams. Typically, an ISO/IEC 14496-17 text stream will be an element of an ISO/IEC 13818-1 program, as defined by the PMT in a transport stream and the PSM in a program stream. The carriage and buffer management of ISO/IEC 14496-17 text streams is defined using existing parameters from Rec. ITU-T H.222.0 | ISO/IEC 13818-1 such as PTS and DTS, as well as information from the ISO/IEC 14496-17 text stream. For this purpose, 2.15.3 specifies the decoding of ISO/IEC 14496-17 streams within the T-STD and P-STD models, using the Hypothetical Text Decoder (HTD) defined in ISO/IEC 14496-17.

When an ISO/IEC 14496-17 text stream is carried in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream, the ISO/IEC 14496-17 coded data shall be contained in PES packets, as defined in 2.15.2. Information needed to decode the ISO/IEC 14496-17 text stream is provided by the MPEG-4 text descriptor, specified in 2.6.70.

Applications that wish to transmit font streams as specified in ISO/IEC 14496-18, can reference the streamed fonts as specified in Annex A.1 of ISO/IEC 14496-17. To transmit ISO/IEC 14496-18 font streams, applications can use the data or object carousel as defined in ISO/IEC 13818-6.

2.15.2 Carriage in PES packets

ISO/IEC 14496-17 text streams are carried in PES packets, using one of the assigned 14 stream_id_extension values. If the textFormat field in TextConfig() is coded with the value 0x01, indicating Timed Text as specified in 3GPP TS 26.245, then the text stream consists of a concatenation of 3GPP Text Access Units in display order, whereby each Text Access Units consists of one or more so-called TTUs. For carriage in PES packets, alignment between PES packets and TTUs is required; i.e., the first byte in the payload of a PES packet carrying an ISO/IEC 14496-17 text stream shall be the first byte of a TTU, which is the first byte of the TTU header specified in 7.4.2 in ISO/IEC 14496-17. There is no further need for alignment; hence, the first TTU in a PES packet may contain for example a non-first fragment of a Text Access unit or a Sample Description.

A PTS shall be coded in the PES packet header of each PES packet carrying an ISO/IEC 14496-17 text stream. The PTS shall refer to the first Text Access Unit that commences in the PES packet. A Text Access Unit commences in a PES packet if a TTU[1] with a complete text sample or the TTU[2] with the first fragment of a text sample is present in the PES packet. For identifying whether a TTU[2] carries the first or a subsequent text sample fragment, see 7.3.2.2 and 7.4.5 in ISO/IEC 14496-17.

2.15.3 STD extensions

2.15.3.1 T-STD extensions

The T-STD model includes a transport buffer TB_n and a multiplex buffer B_n prior to the Hypothetical Text Decoder defined in ISO/IEC 14496-17. transport stream packets with ISO/IEC 14496-17 data, as indicated by its PID, enter the buffer TB_n. Bytes are removed from TB_n to enter Buffer B_n at the rate Rx_n. Delivery of bytes from B_n to enter the TTU Decoder in the HTD is at rate Rbx_n. See also Figure 2-17.

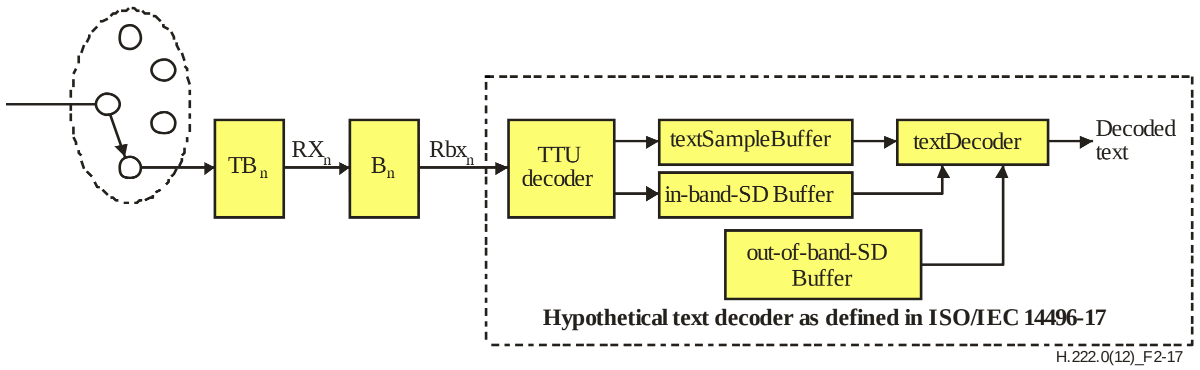


Figure 2-17 – T-STD model extensions for ISO/IEC 14496-17 text streams

Buffer management

The size TBS_n of TB_n is equal to 512 bytes. For ISO/IEC 14496-17 text streams, the size BS_n of B_n is equal to 4096 bytes. The rate Rx_n is equal to 2 000 000 bits/s. The rate Rbx_n meets the delivery schedule defined for ISO/IEC 14496-17 data at the input of the HTD in clause 7.7 of ISO/IEC 14496-17, i.e., Rbx_n = R[profile, level] if the textSampleBuffer is not full and Rbx_n = 0 if the textSampleBuffer is full, where R[profile, level] is defined as the profile and level specific rate R in clause 7.8 of ISO/IEC 14496-17.

Each of the following requirements shall apply:

– Buffer TB_n shall not overflow.
– Buffer B_n shall not overflow.
– Each HTD requirement specified in clause 7.7 of ISO/IEC 14496-17 shall be met.

2.15.3.2 P-STD extensions

The P-STD model for the decoding of an ISO/IEC 14496-17 text stream includes a multiplex buffer B_n prior to the Hypothetical Text Decoder defined in ISO/IEC 14496-17. For each ISO/IEC 14496-17 text stream n, the size BS_n of buffer B_n in the P-STD is defined by the P-STD_buffer_size field in the PES packet header.

Each of the following requirements shall apply:

– Buffer B_n shall not overflow.

– Each HTD requirement specified in clause 7.7 of ISO/IEC 14496-17 shall be met.

2.16 Carriage of auxiliary video streams

ISO/IEC 23002-3 specifies auxiliary video streams. ISO/IEC 23002-3 auxiliary video streams can be carried over Rec. ITU-T H.222.0 | ISO/IEC 13818-1 streams as follows:

- in Table 2-27, 16 stream-id_extension values are assigned to signal auxiliary video streams;
- in Table 2-34, one stream-type value is assigned to signal an auxiliary video stream;
- in Table 2-45, one descriptor tag is assigned to indicate an auxiliary video stream descriptor;
- in 2.6.74 the auxiliary video stream descriptor is specified;
- the auxiliary video stream descriptor is associated with each auxiliary video stream.

Auxiliary video streams provide additional information about a conventional primary video sequence, as specified in ISO/IEC 23002-3. The auxiliary video stream shall be synchronized with its primary video counterpart through the use of timestamps in the associated PES header based on the same PCR clock.

In case a program contains multiple video streams, it will be up to the application to specify the association between the video component and auxiliary video streams.

2.17 Carriage of HEVC

2.17.1 Constraints for the transport of HEVC

For HEVC video streams, HEVC temporal video sub-bitstreams or HEVC temporal video subsets, the following constraints additionally apply:

- Each HEVC access unit shall contain an access unit delimiter NAL unit.

  NOTE 1 – HEVC requires that an access unit delimiter NAL unit, if present, is the first NAL unit within an HEVC access unit. Access unit delimiter NAL units simplify the ability to detect the boundary between HEVC access units.

- An HEVC video stream or HEVC temporal video sub-bitstream of an HEVC video stream conforming to one or more profiles defined in Annex A of Rec. ITU-T H.265 | ISO/IEC 23008-2 shall be an element of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program and the stream_type for this elementary stream shall be equal to 0x24.

  NOTE 2 – Such a stream can be the HEVC base sub-partition of an HEVC video stream conforming to one or more profiles defined in Annex G or Annex H of Rec. ITU-T H.265 | ISO/IEC 23008-2.

- The video parameter sets, sequence parameter sets and picture parameter sets, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2, that are necessary for decoding an HEVC video stream or HEVC temporal video sub-bitstream shall be present within the elementary stream carrying that HEVC video stream or HEVC temporal video sub-bitstream.

- For each HEVC temporal video subset of an HEVC video stream conforming to one or more profiles defined in Annex A of Rec. ITU-T H.265 | ISO/IEC 23008-2 that is an element of the same Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, the stream_type for this elementary stream shall be equal to 0x25.

  NOTE 3 – Such a stream can be an HEVC sub-partition of an HEVC video stream conforming to one or more profiles defined in Annex G or Annex H of Rec. ITU-T H.265 | ISO/IEC 23008-2.

- An HEVC enhancement sub-partition of an HEVC video stream conforming to one or more profiles defined in Annex G or Annex H of Rec. ITU-T H.265 | ISO/IEC 23008-2 shall be an element of an Rec. ITU-T H.222.0 | ISO/IEC 13818 1 program. The stream_type for this elementary stream shall be set according to Table 2-34.

- The video parameter sets, sequence parameter sets, and picture parameter sets, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2, necessary for decoding an HEVC video stream, HEVC temporal video sub-bitstream, or HEVC base sub-partition shall be present within the elementary stream carrying that HEVC video stream, HEVC temporal video sub-bitstream, or HEVC base sub-partition.

- When a Rec. ITU-T H.222.0 | ISO/IEC 13818 1 program includes one or more elementary streams with stream_type equal to 0x28, 0x29, 0x2A or 0x2B, at least one HEVC operation point descriptor shall be present in the program map table associated with the program.
- When a Rec. ITU-T H.222.0 | ISO/IEC 13818 1 program includes more than one elementary stream with the same stream_type value of 0x24, 0x25 or in the range of 0x28-0x2B and hierarchy cannot be implied as specified in Table 2-137, one hierarchy descriptor as defined in 2.6.7 shall be present for each elementary stream with a stream_type value of 0x24, 0x25 and one HEVC hierarchy extension descriptor as defined in 2.6.102 shall be present for each elementary stream with a stream_type value in the range of 0x28-0x2B.

  NOTE 4 – Hierarchy descriptors or HEVC hierarchy extension descriptors are needed to assign a hierarchy layer index to each elementary stream if hierarchy cannot be implied, as specified in Table 2-137.
- In each elementary stream with *stream_type* equal to 0x24 with a hierarchy descriptor, the *hierarchy_type* in the hierarchy descriptor shall be equal to 15.
- In each elementary stream with *stream_type* equal to 0x25 with a hierarchy descriptor, the *hierarchy_type* in the hierarchy descriptor shall be equal to 3.
- The video parameter sets, sequence parameter sets and picture parameter sets, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2, that are necessary for decoding the HEVC highest temporal sub-layer representation of an HEVC temporal video subset shall be present within the elementary stream carrying the HEVC temporal video sub-bitstream associated by a hierarchy descriptor.
- The aggregation of the HEVC temporal video sub-bitstream with associated HEVC temporal video subsets according to the hierarchy descriptors, as specified in 2.17.3, shall result in a valid HEVC video stream.

  NOTE 5 – The resulting HEVC video stream contains a set of temporal sub-layers, as specified in Rec. ITU-T H.265 | ISO/IEC 23008-2, with TemporalId values forming a contiguous range of integer numbers.
- The aggregation, as specified in 2.17.4, of an HEVC enhancement sub-partition with all HEVC sub-partitions according to the HEVC operation signalled in the HEVC operation point descriptor shall result in a valid layered HEVC video stream.

  NOTE 6 – The resulting HEVC video stream is the HEVC operation point of that HEVC enhancement sub-partition.
- Each HEVC picture with nuh_layer_id larger than 0 shall be contained within an elementary stream with a stream_type equal to either 0x28, 0x29, 0x2A or 0x2B.
- An elementary stream ES_{te} with stream_type equal to either 0x29 or 0x2B shall satisfy the following:
  - The elementary stream ES_{te} contains an HEVC temporal video subset which is a temporal enhancement for exactly one reference elementary stream ES_{ref}, i.e., the layer L_i present in the reference elementary stream ES_{ref} shall also be present in the elementary stream ES_{te}.

    NOTE 7 – The reference elementary stream ES_{ref} is either an HEVC temporal video sub-bitstream with a stream_type equal to either 0x28 or 0x2A or another HEVC temporal video subset with stream_type equal to either 0x29 or 0x2B, respectively, for which the same applies.

Carriage in PES packets

Rec. ITU-T H.265 | ISO/IEC 23008-2 video is carried in PES packets as PES_packet_data_bytes, using one of the 16 stream_id values assigned to video, while signalling the Rec. ITU-T H.265 | ISO/IEC 23008-2 video stream, by means of the assigned stream-type value in the PMT (see Table 2-34). The highest level that may occur in an HEVC video stream, as well as a profile and tier that the entire stream conforms to should be signalled using the HEVC video descriptor. If an HEVC video descriptor is associated with an HEVC video stream, an HEVC temporal video sub-bitstream, an HEVC temporal video subset or an HEVC enhancement sub-partition, then this descriptor shall be conveyed in the descriptor loop for the respective elementary stream entry in the program map table. This Recommendation | International Standard does not specify the presentation of Rec. ITU-T H.265 | ISO/IEC 23008-2 streams in the context of a program stream.

For PES packetization, no specific data alignment constraints apply. For synchronization and STD management, PTSs and, when appropriate, DTSs are encoded in the header of the PES packet that carries the ITU-T H.265 | ISO/IEC 23008-2 video elementary stream data. For PTS and DTS encoding, the constraints and semantics apply as defined in 2.4.3.7 and 2.7.

DPB buffer management

Carriage of an HEVC video stream, an HEVC temporal video sub-stream, an HEVC temporal video subset or an HEVC enhancement sub-partition over Rec. ITU-T H.222.0 | ISO/IEC 13818-1 does not impact the size of the buffer DPB. For decoding of an HEVC video stream, an HEVC temporal video sub-bitstream or an HEVC temporal video sub-bitstream and its associated HEVC temporal video subsets in the STD, the size of DPB is as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2. The DPB shall be managed as specified in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2 (clauses C.3 and C.5). A decoded HEVC access unit enters the DPB instantaneously upon decoding the HEVC access unit, hence at the CPB removal time of the HEVC access unit. A decoded HEVC access unit is presented at the DPB output time. If the HEVC video stream, HEVC temporal video sub-bitstream or HEVC temporal video subset provides insufficient information to determine the CPB removal time and the DPB output time of HEVC access units, then these time instants shall be determined in the STD model from PTS and DTS timestamps as follows:

1) The CPB removal time of HEVC access unit n is the instant in time indicated by DTS(n) where DTS(n) is the DTS value of HEVC access unit n.

2) The DPB output time of HEVC access unit n is the instant in time indicated by PTS(n) where PTS(n) is the PTS value of HEVC access unit n.

NOTE 8 – HEVC video sequences in which the *low_delay_hrd_flag* in the syntax structure *hrd_parameters()* is set to 1 carry sufficient information to determine the DPB output time and the CPB removal time of each HEVC access unit. Hence for HEVC access units for which STD underflow may occur, the CPB removal time and the DPB output time are defined by HRD parameters, and not by DTS and PTS timestamps.

NOTE 9 – An HEVC video stream may carry information to determine compliance of the HEVC video stream to the HRD, as specified in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2. The presence of this information can be signalled in a transport stream using the HEVC timing and HRD descriptor with the *hrd_management_valid_flag* set to '1'. Irrespective of the presence of this information, compliance of an HEVC video stream to the T-STD ensures that HRD buffer management requirements for the CPB are met when each byte in the HEVC video stream is delivered to and removed from the CPB in the HRD at exactly the same instant in time at which the byte is delivered to and removed from EB_n in the T-STD.

2.17.2 T-STD Extensions for single layer HEVC

When there is an HEVC video stream or HEVC temporal video sub-bitstream in an ITU-T H.222.0 | ISO/IEC 13818-1 program and there is no HEVC temporal video subset associated with this elementary stream of stream_type 0x24 in the same ITU-T H.222.0 | ISO/IEC 13818-1 program, the T-STD model as described in 2.4.2.1 is extended as illustrated in Figure 2-18 and as specified below.

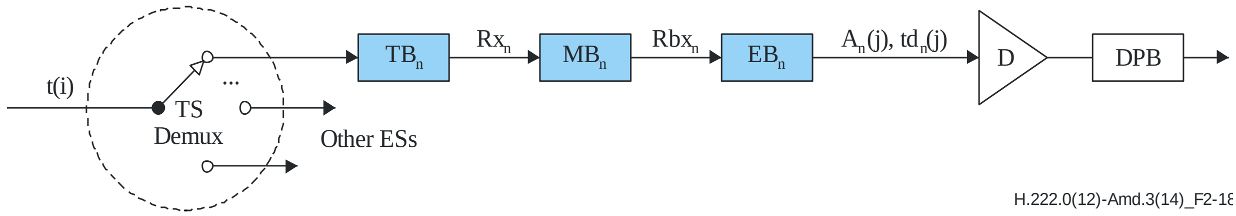


Figure 2-18 – T-STD model extensions for single layer HEVC

TB_n, MB_n, EB_n buffer management

The following additional notations are used to describe the T-STD extensions and are illustrated in Figure 2-18.

t(i) indicates the time in seconds at which the i-th byte of the transport stream enters the system target decoder

TB_n is the transport buffer for elementary stream n

TBS is the size of the transport buffer TB_n, measured in bytes

MB_n is the multiplexing buffer for elementary stream n

MBS_n is the size of the multiplexing buffer MB_n, measured in bytes

EB_n is the elementary stream buffer for the HEVC video stream

j is an index to the HEVC access unit of the HEVC video stream

$A_n(j)$ is the j-th access unit of the HEVC video bitstream

$td_n(j)$ is the decoding time of $A_n(j)$, measured in seconds, in the system target decoder

Rx_n is the transfer rate from the transport buffer TB_n to the multiplex buffer MB_n as specified below.

Rbx_n is the transfer rate from the multiplex buffer MB_n to the elementary stream buffer EB_n as specified below.

The following apply:

- There is exactly one transport buffer TB_n for the received HEVC video stream or HEVC temporal video sub-bitstream where the size TBS is fixed to 512 bytes.

- There is exactly one multiplexing buffer MB_n for the HEVC video stream or HEVC temporal video sub-bitstream, where the size MBS_n of the multiplexing buffer MB is constrained as follows:

  $MBS_n = BS_{mux} + BS_{oh} + CpbNalFactor \times MaxCPB[tier, level] – cpb\_size$

  where BS_{oh}, packet overhead buffering, is defined as:

  $BS_{oh} = (1/750)$ seconds × max{ BrNalFactor × MaxBR[tier, level], 2 000 000 bit/s}

  and BS_{mux}, additional multiplex buffering, is defined as:

  $BS_{mux} = 0.004$ seconds × max{ BrNalFactor × MaxBR[tier, level], 2 000 000 bit/s}

  MaxCPB[tier, level] and MaxBR[tier, level] are taken from Annex A of Rec. ITU-T H.265 | ISO/IEC 23008-2 for the tier and level of the HEVC video stream or HEVC temporal video sub-bitstream where rates are expressed in bit/s. cpb_size is taken from the HRD parameters, as specified in Annex E of Rec. ITU-T H.265 | ISO/IEC 23008-2, included in the HEVC video stream or HEVC temporal video sub-bitstream, where the size is expressed in bits. Implicit conversion is carried out according to Note 2 in 2.4.2.4.

- There is exactly one elementary stream buffer EB_n for all the elementary streams in the set of received elementary streams associated by hierarchy descriptors, with a total size EBS_n

  $$EBS_n = cpb\_size$$

  where cpb_size is taken from the HRD parameters, as specified in Annex E of Rec. ITU-T H.265 | ISO/IEC 23008-2, included in the HEVC video stream or the HEVC temporal video sub-bitstream, where the size is expressed in bits. Implicit conversion is carried out according to Note 2 in 2.4.2.4.

- Transfer from TB_n to MB_n is applied as follows:

  When there is no data in TB_n then Rx_n is equal to zero. Otherwise:

  $$Rx_n = bit\_rate$$

  where bit_rate is BrNalFactor/BrVlcFactor x BitRate[SchedSelIdx] of data flow into the CPB for the byte stream format and BitRate[SchedSelIdx] is as defined in Annex E of Rec. ITU-T H.265 | ISO/IEC 23008-2 when NAL hrd_parameters() is present in the VUI parameters of the HEVC video stream.

  NOTE – Annex E also specifies default values for BitRate[SchedSelIdx] based on profile, tier and level when NAL HRD parameters are not present in the VUI.

- Transfer from MB_n to EB_n is applied as follows:

  If the *HEVC_timing_and_HRD_descriptor* is present with the *hrd_management_valid_flag* set to '1' for the elementary stream, then the transfer of data from MB_n to EB_n shall follow the HRD defined scheme for data arrival in the CPB of the elementary stream as defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2.

  Otherwise, the leak method shall be used to transfer data from MB_n to EB_n as follows:

  $$Rbx_n = BrNalFactor \times MaxBR[tier, level]$$

  where MaxBR[tier, level] is taken from Annex A of Rec. ITU-T H.265 | ISO/IEC 23008-2 for the tier and level of the HEVC video stream or HEVC temporal video sub-bitstream.

  If there is PES packet payload data in MB_n, and buffer EB_n is not full, the PES packet payload is transferred from MB_n to EB_n at a rate equal to Rbx_n. If EB_n is full, data are not removed from MB_n. When a byte of data is transferred from MB_n to EB_n, all PES packet header bytes that are in MB_n and precede that byte are instantaneously removed and discarded. When there is no PES packet payload data present in MB_n, no data is removed from MB_n. All data that enters MB_n leaves it. All PES packet payload data bytes enter EB_n instantaneously upon leaving MB_n.

STD delay

The STD delay of any ITU-T H.265 | ISO/IEC 23008-2 data other than HEVC still picture data through the system target decoders buffers TB_n, MB_n, and EB_n shall be constrained by $td_n(j) – t(i) \leq 10$ seconds for all j, and all bytes i in access unit $A_n(j)$.

The delay of any HEVC still picture data through the system target decoders TB_n, MB_n, and EB_n shall be constrained by $td_n(j) – t(i) \leq 60$ seconds for all j, and all bytes i in access unit $A_n(j)$.

Buffer management conditions

Transport streams shall be constructed so that the following conditions for buffer management are satisfied:

- Each TB_n shall not overflow and shall be empty at least once every second.
- Each MB_n, EB_n and DPB shall not overflow.
- EB_n shall not underflow, except when VUI parameters are present for the HEVC video sequence with the *low_delay_hrd_flag* set to '1'. Underflow of EB_n occurs for HEVC access unit $A_n(j)$ when one or more bytes of $A_n(j)$ are not present in EB_n at the decoding time $td_n(j)$.

2.17.3 T-STD extensions for layered transport of HEVC temporal video subsets

When there is an HEVC video sub-bitstream and at least one associated elementary stream of type 0x25 in an ITU-T H.222.0 | ISO/IEC 13818-1 program, the T-STD model as described in 2.4.2.1 is extended as illustrated in Figure 2-19 and as specified below.

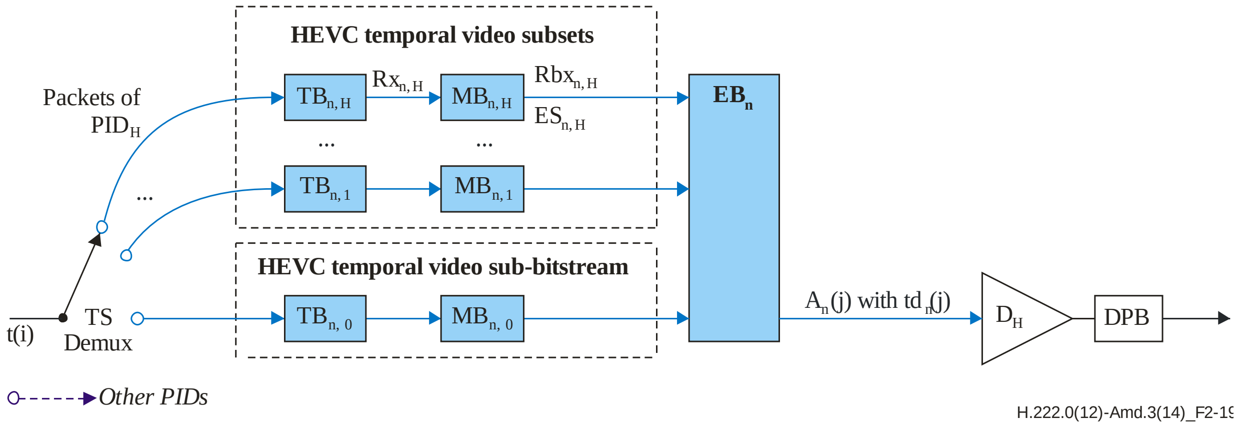


Figure 2-19 – T-STD model extensions for layered transport of HEVC temporal video subsets

The following additional notations are used to describe the T-STD extensions and are illustrated in Figure 2-19.

$t(i)$ indicates the time in seconds at which the i-th byte of the transport stream enters the system target decoder.

H is the number of received HEVC temporal video subsets, associated by hierarchy descriptors with the same HEVC temporal video sub-bitstream.

k is an index identifying the H+1 received elementary streams which contain exactly one HEVC temporal video sub-bitstream and H HEVC temporal video subsets associated by hierarchy descriptors. The index value k equal to 0 identifies the elementary stream which contains the HEVC temporal video sub-bitstream and index values k ranging from 1 up to H identify the associated HEVC temporal video subsets.

$ES_{n,k}$ is the received elementary stream which contains the k-th HEVC temporal video subset or the HEVC temporal video sub-bitstream if k equals 0.

$ES_{n,H}$ is the received elementary stream containing the highest HEVC temporal video subset present in the set of received elementary streams.

PID_H is the packet identifier value which identifies $ES_{n,H}$.

j is an index to the output access units.

$A_n(j)$ is the j-th access unit of the HEVC complete temporal representation.

$td_n(j)$ is the decoding time of $A_n(j)$ in the system target decoder.

$TB_{n,k}$ is the transport buffer for elementary stream k.

$TBS_{n,k}$ is the size of the transport buffer $TB_{n,k}$, measured in bytes.

$MB_{n,k}$ is the multiplexing buffer for elementary stream k.

$MBS_{n,k}$ is the size of the multiplexing buffer $MB_{n,k}$, measured in bytes.

EB_n is the elementary stream buffer for the received HEVC temporal video sub-bitstream $ES_{n,0}$ and the received HEVC temporal video subsets $ES_{n,1}$ to $ES_{n,H}$.

EBS_n is the size of elementary stream buffer EB_n, measured in bytes.

$Rx_{n,k}$ is the transfer rate from the k-th transport buffer $TB_{n,k}$ to the k-th multiplex buffer $MB_{n,k}$ as specified below.

$Rbx_{n,k}$ is the transfer rate from the k-th multiplex buffer $MB_{n,k}$ to the elementary stream buffer EB_n as specified below.

NOTE – The index n, where used, indicates that the received elementary streams and associated buffers belong to a certain HEVC temporal video sub-bitstream and its associated HEVC temporal video subsets, distinguishing these elementary streams and associated buffers from other elementary streams and buffers, maintaining consistency with the notation in Figure 2-18.

$TB_{n,k}$, $MB_{n,k}$, EB_n buffer management

The following apply:

- There is one transport buffer $TB_{n,k}$ for each received elementary stream $ES_{n,k}$, where the size $TBS_{n,k}$ is fixed to 512 bytes.
- There is one multiplex buffer $MB_{n,k}$ for each received elementary stream $ES_{n,k}$, where the size $MBS_{n,k}$ of the multiplex buffer $MB_{n,k}$ is constrained as follows:

  $$MBS_{n,k} = BS_{mux} + BS_{oh} + CpbNalFactor \times MaxCPB[tier, level] – cpb\_size$$

  where

  BS_{oh}, packet overhead buffering, and BS_{mux}, additional multiplex buffering, are as specified in 2.17.2;

  MaxCPB[tier, level] and MaxBR[tier, level] are taken from the tier and level specification of HEVC for the tier and level of the HEVC highest temporal sub-layer representation associated with $ES_{n,k}$;

  cpb_size is taken from the HRD parameters, as specified in Annex E of Rec. ITU-T H.265 | ISO/IEC 23008-2, included in the HEVC highest temporal sub-layer representation associated with $ES_{n,k}$. In the HRD parameters, cpb_size is expressed in bits, and its value is implicitly converted into a value expressed in bytes according to Note 2 in 2.4.2.4.
- There is exactly one elementary stream buffer EB_n for the H + 1 elementary streams in the set of received elementary streams $ES_{n,0}$ to $ES_{n,H}$, with a total size EBS_n

  $$EBS_n = cpb\_size$$

  where cpb_size is taken from the HRD parameters, as specified in Annex E of Rec. ITU-T H.265 | ISO/IEC 23008-2, included in the HEVC highest temporal sub-layer representation associated with $ES_{n,H}$. In the HRD parameters, cpb_size is expressed in bits, and its value is implicitly converted into a value expressed in bytes according to Note 2 in 2.4.2.4.
- Transfer from $TB_{n,k}$ to $MB_{n,k}$ is applied as follows:

  When there is no data in $TB_{n,k}$ then $Rx_{n,k}$ is equal to zero. Otherwise:

  $$Rx_{n,k} = bit\_rate$$

  where bit_rate is as specified in 2.17.2.
- Transfer from $MB_{n,k}$ to EB_n is applied as follows:

  If the HEVC_timing_and_HRD_descriptor is present with the hrd_management_valid_flag set to '1' for the HEVC video sub-bitstream, then the transfer of data from $MB_{n,k}$ to EB_n shall follow the HRD defined scheme for data arrival in the CPB of elementary stream $ES_{n,H}$ as defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2.

  Otherwise, the leak method shall be used to transfer data from $MB_{n,k}$ to EB_n as follows:

  $$Rbx_{n,k} = BrNalFactor \times MaxBR[tier, level]$$

  where MaxBR[tier, level] is defined for the byte stream format in Annex A of Rec. ITU-T H.265 | ISO/IEC 23008-2 for the tier and level of the HEVC video stream or the HEVC highest temporal sub-layer representation associated with $ES_{n,k}$.

  If there is PES packet payload data in $MB_{n,k}$, and EB_n is not full, the PES packet payload is transferred from $MB_{n,k}$ to EB_n at a rate equal to $Rbx_{n,k}$. If EB_n is full, data are not removed from $MB_{n,k}$. When a byte of data is transferred from $MB_{n,k}$ to EB_n, all PES packet header bytes that are in $MB_{n,k}$ and precede that byte are instantaneously removed and discarded. When there is no PES packet payload data present in $MB_{n,k}$, no data is removed from $MB_{n,k}$. All data that enters $MB_{n,k}$ leaves it. All PES packet payload data bytes enter EB_n instantaneously upon leaving $MB_{n,k}$.

At the output of the elementary stream buffer EB_n, the elementary streams are aggregated by removing all HEVC access units in ascending DTS order and transferring them to the HEVC decoder D_H, irrespective of which elementary stream $ES_{n,k}$ each HEVC access unit belongs to.

STD delay

The STD delay of any ITU-T H.265 | ISO/IEC 23008-2 data other than HEVC still picture data through the system target decoders buffers $TB_{n,k}$, $MB_{n,k}$, and EB_n shall be constrained by $td_n(j) – t(i) \leq 10$ seconds for all k, all j, and all bytes i in access unit $A_n(j)$.

The delay of any HEVC still picture data through the system target decoders $TB_{n,k}$, $MB_{n,k}$, and EB_n shall be constrained by $td_n(j) – t(i) \leq 60$ seconds for all k, all j, and all bytes i in access unit $A_n(j)$.

Buffer management conditions

Transport streams shall be constructed so that the following conditions for buffer management are satisfied:

- Each $TB_{n,k}$ shall not overflow and shall be empty at least once every second.
- Each $MB_{n,k}$, EB_n, and DPB shall not overflow.

• EB_n shall not underflow, except when VUI parameters are present for the HEVC video sequence with the *low_delay_hrd_flag* set to '1'. Underflow of EB_n occurs for HEVC access unit $A_n(j)$ when one or more bytes of $A_n(j)$ are not present in EB_n at the decoding time $td_n(j)$.

2.17.4 T-STD extensions for layered transport of HEVC sub-partitions with bitstream-partition-specific CPB operation

When there is at least one elementary stream with stream_type value in the range of 0x28 to 0x2B in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, the T-STD model as described in 2.4.2.1 is extended for elementary streams with a stream_type value in the range of 0x28 to 0x2B as illustrated in Figure 2-20 and as specified below.

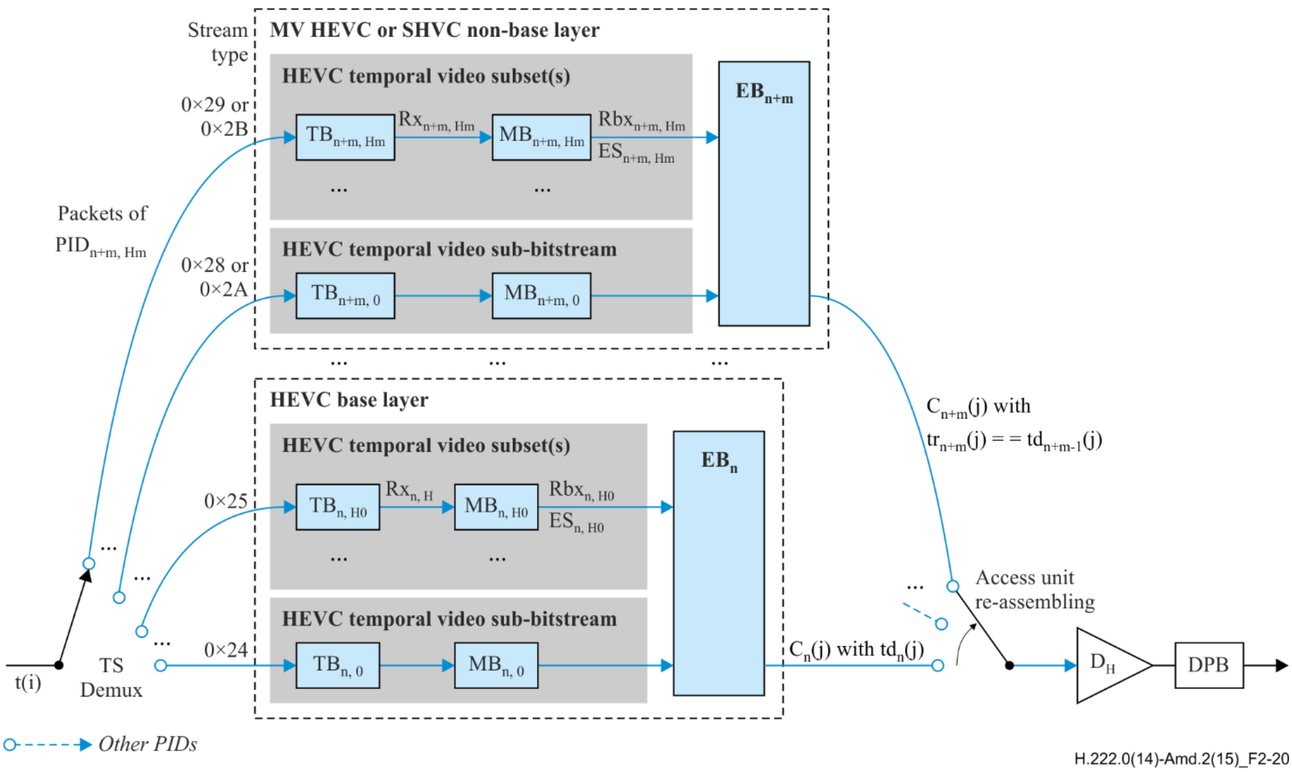


Figure 2-20 – T-STD model extensions for bitstream-partition-specific CPB operation

NOTE – The lower dashed box containing blocks that handle the HEVC base layer represents the T-STD buffer model as specified in 2.17.2, if there are no HEVC temporal video subsets, or 2.17.3, if the program contains at least one program element with stream_type equal to 0x25.

The following additional notations are used to describe the T-STD extensions and are illustrated in Figure 2-20 above.

t(i) Indicates the time in seconds at which the i-th byte of the transport stream enters the system target decoder.

l Is an index into the received HEVC sub-partitions of stream_type 0x28 or 0x2A (which include TemporalId 0). The order of HEVC sub-partitions is indicated by the HEVC operation point descriptor. The same index also applies to corresponding HEVC temporal enhancement sub-partitions. Here, l starts from n, which is associated with the HEVC base sub-partition, and runs up to (n+m), where m is specified below.

m Is the number of received HEVC sub-partitions of stream_type 0x28 or 0x2A.

Hl Is the number of received HEVC corresponding temporal enhancement sub-partitions of the l-th received HEVC sub-partition of stream_type 0x28 or 0x2A, associated by HEVC hierarchy extension descriptors with the same HEVC base sub-partition.

$ES_{l,k}$ Is the received elementary stream which contains the k-th HEVC corresponding temporal enhancement sub-partition of the l-th received HEVC sub-partition of stream_type 0x28 or 0x2A, or the l-th HEVC sub-partition of stream_type 0x28 or 0x2A if k equals 0.

$ES_{n+m,Hm}$ Is the received elementary stream which contains the HEVC sub-partition of the highest HEVC operation point in the set of received elementary streams.

$PID_{n+m,Hm}$ Is the packet identifier value which identifies $ES_{n+m,Hm}$.

j Is an index to the output HEVC access units.

$C_l(j)$ Is the j-th HEVC layer component of the l-th received HEVC sub-partition of stream_type 0x28 or 0x2A or HEVC corresponding temporal enhancement sub-partition.

$A_n(j)$ Is the j-th HEVC access unit of the HEVC complete temporal representation.

$td_n(j)$ Is the decoding time of $A_n(j)$ in the system target decoder.

$tr_l(j)$ Is the value of TREF, if available in the PES header attached to $C_l(j)$, else the decoding time of $A_n(j)$ in the system target decoder.

$TB_{l,k}$ Is the transport buffer for elementary stream $ES_{l,k}$.

$TBS_{l,k}$ Is the size of the transport buffer $TB_{l,k}$, measured in bytes.

$MB_{l,k}$ Is the multiplexing buffer for elementary stream $ES_{l,k}$.

$MBS_{l,k}$ Is the size of the multiplexing buffer $MB_{l,k}$, measured in bytes.

EB_l Is the elementary stream buffer for the received HEVC temporal video sub-bitstream $ES_{l,0}$ and the received HEVC temporal video subsets $ES_{l,1}$ to $ES_{l,H}$.

NOTE 1 – Each buffer EB_l contains one partition as specified in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2.

EBS_l Is the size of elementary stream buffer EB_l, measured in bytes.

$Rx_{l,k}$ Is the transfer rate from the k-th transport buffer $TB_{l,k}$ to the k-th multiplex buffer $MB_{l,k}$ as specified below.

$Rbx_{l,k}$ Is the transfer rate from the k-th multiplex buffer $MB_{l,k}$ to the elementary stream buffer EB_l as specified below.

NOTE 2 – The index n, where used, indicates that the received elementary streams and associated buffers belong to a certain HEVC base sub-partition, distinguishing these elementary streams and associated buffers from other elementary streams and buffers, maintaining consistency with the notation in Figure 2-1 and other T-STD extensions.

$TB_{l,k}$, $MB_{l,k}$, EB_l buffer management

The following applies:

- There is one transport buffer $TB_{l,k}$ for each received elementary stream $ES_{l,k}$, where the size $TBS_{l,k}$ is fixed to 512 bytes.
- There is one multiplex buffer $MB_{l,k}$ for each received elementary stream $ES_{l,k}$, where the size $MBS_{l,k}$ of the multiplex buffer $MB_{l,k}$ is constrained as follows:

  $MBS_{l,k} = BS_{mux} + BS_{oh} + CpbNalFactor \times MaxCPB[tier, level] - cpb\_size$

where

BS_{oh}, packet overhead buffering, and BS_{mux}, additional multiplex buffering, are as specified in 2.17.2;

MaxCPB[tier, level] and MaxBR[tier, level] are taken from the tier and level specification of the HEVC for the tier and level of $ES_{l,k}$ the HEVC operation point associated with $ES_{l,k}$;

cpb_size is taken from the sub-layer HRD parameters within the applicable hrd_parameters( ), as specified in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, for the HEVC operation point associated with $ES_{l,k}$. In the HRD parameters, cpb_size is expressed in bits, and its value is implicitly converted into a value expressed in bytes according to Note 2 in 2.4.2.4.

- There is one elementary stream buffer EB_l for the Hl + 1 elementary streams in the set of received elementary streams $ES_{l,0}$ to $ESl_{m,Hl}$, with a total size EBS_l

EBS_l = cpb_size

where cpb_size is taken from the sub-layer HRD parameters within the applicable hrd_parameters( ), as specified in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, for the HEVC operation point associated with $ES_{l,H}$. In the HRD parameters, cpb_size is expressed in bits, and its value is implicitly converted into a value expressed in bytes according to Note 2 in 2.4.2.4.

- Transfer from $TB_{l,k}$ to $MB_{l,k}$ is applied as follows:
  – When there is no data in $TB_{l,k}$ then $Rx_{l,k}$ is equal to zero.
  – Otherwise, $Rx_{l,k}$ = bit_rate

bit_rate is taken from the NAL HRD parameters within the applicable hrd_parameters( ), as specified in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, for the HEVC operation point associated with $ES_{l,k}$, if sub-layer HRD parameters are present in the VPS and nal_hrd_parameters_present_flag is set to '1'.

Otherwise:

bit_rate = BrNalFactor / BrVclFactor × $BitRate_{VCL}$, if sub-layer HRD parameters within the applicable hrd_parameters( ), as specified in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, for the HEVC operation point associated with $ES_{l,k}$ are present in the VPS and vcl_hrd_parameters_present_flag is set to '1'; $BitRate_{VCL}$ is taken from the VCL HRD parameters.

BrNalFactor and BrVclFactor are as defined in Rec. ITU-T H.265 | ISO/IEC 23008-2 for the profile, tier, and level of $ES_{l,k}$ the HEVC operation point associated with $ES_{l,k}$.

In both cases, the index of the applicable hrd_parameters( ) syntax structure in the active VPS of the HEVC video stream is given by: bsp_hrd_idx[target_ols][0][max_temporal_id][target_schedule_idx][l + 1] as specified in Annex F of Rec. ITU-T H.265 | ISO/IEC 23008-2, where target_ols and max_temporal_id are signalled in the HEVC operation point descriptor for the HEVC operation point associated with $ES_{l,k}$, and target_schedule_idx is signalled in the HEVC timing and HRD descriptor.

- Transfer from $MB_{l,k}$ to EB_l is applied as follows:
  – If the HEVC_timing_and_HRD_descriptor is present with the hrd_management_valid_flag set to '1' for the HEVC video sub-bitstream, then the transfer of data from $MB_{l,k}$ to EB_l shall follow the HRD defined scheme for data arrival in the CPB of elementary stream $ES_{l,H}$ as defined in Annex C of Rec. ITU-T H.265 | ISO/IEC 23008-2.
  – Otherwise, the leak method shall be used to transfer data from $MB_{l,k}$ to EB_l as follows:

$Rbx_{n,k}$ = BrNalFactor × MaxBR[tier, level]

where MaxBR[tier, level] is as defined for the byte stream format in the tier and level specification of Rec. ITU T H.265 | ISO/IEC 23008-2 (Table A.2) for the tier and level for $ES_{l,k}$ signalled in the HEVC operation point descriptor for $ES_{l,k}$.

If there is PES packet payload data in $MB_{l,k}$, and EB_l is not full, the PES packet payload is transferred from $MB_{l,k}$ to EB_l at a rate equal to $Rbx_{l,k}$. If EB_l is full, data are not removed from $MB_{l,k}$. When a byte of data is transferred from $MB_{l,k}$ to EB_l, all PES packet header bytes that are in $MB_{l,k}$ and precede that byte are instantaneously removed and discarded. When there is no PES packet payload data present in $MB_{l,k}$, no data is removed from $MB_{l,k}$. All data that enters $MB_{l,k}$ leaves it. All PES packet payload data bytes enter EB_l instantaneously upon leaving $MB_{l,k}$.

Aggregation of elementary streams

The HEVC layer list for an HEVC operation point is the OperationPointESList[] associated with the HEVC operation point.

When there is no hierarchy descriptor or HEVC hierarchy extension descriptor present in the program map table associated with an Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program, a value for the hierarchy_layer_index is implicitly assigned for each elementary stream with stream_type 0x24, 0x25, 0x28, 0x29, 0x2A and 0x2B as described in Table 2-137.

Table 2-137 – Implied hierarchy_layer_index if no hierarchy descriptors are used

Existing stream types	Implied hierarchy_layer_index for program element with stream_type value					
	0x24	0x25	0x28	0x29	0x2A	0x2B
0x24	0					
0x24, 0x25	0	1				
0x24, 0x28	0		1			
0x24, 0x25, 0x28	0	1	2			
0x24, 0x25, 0x28, 0x29	0	1	2	3		
0x24, 0x2A	0				1	
0x24, 0x2A, 0x2B	0				1	2
0x24, 0x25, 0x2A	0	1			2	
0x24, 0x25, 0x2A, 0x2B	0	1			2	3

The HEVC operation point is aggregated from the HEVC layer components at the output of the elementary stream buffers EB_l by determining the value of TREF, if available in the PES header, or else TREF is set to DTS, for the next HEVC layer component at the output of EB_{n+m}, and gathering all HEVC layer components with a DTS equal to TREF, in the order given by the HEVC layer list as specified above, and transferring them to the HEVC decoder D_H.

Carriage in PES packets

For correct re-assembling of the HEVC layer components to an HEVC access unit, if there is an HEVC dependency representation in the same HEVC access unit in more than one elementary stream, the following applies:

- Each PES packet shall contain exactly one HEVC layer component;
- The PTS and, if applicable, the DTS value shall be provided in the PES header of each HEVC layer component;
- If the DTS value of the HEVC layer component in an elementary stream is different from the DTS value of the HEVC layer component of the same HEVC access unit in an ES listed in the hierarchy descriptor or the HEVC hierarchy extension descriptors of the first elementary stream, the TREF field as defined in 2.4.3.7 shall be present in the PES header extension of the HEVC layer component of first elementary stream and the TREF field value shall be equal to the DTS value of the HEVC layer component of the second elementary stream.

STD delay

The STD delay of any Rec. ITU-T H.265 | ISO/IEC 23008-2 data other than HEVC still picture data through the System Target Decoders buffers $TB_{l,k}$, $MB_{l,k}$, and EB_l shall be constrained by $td_l(j) - t(i) \leq 10$ seconds for all l, all k, all j, and all bytes i in HEVC access unit $A_n(j)$.

The delay of any HEVC still picture data through the system target decoders $TB_{l,k}$, $MB_{l,k}$, and EB_l shall be constrained by $td_l(j) - t(i) \leq 60$ seconds for all l. all k, all j, and all bytes i in HEVC access unit $A_n(j)$.

Buffer management conditions

Transport streams shall be constructed so that the following conditions for buffer management are satisfied:

- Each $TB_{l,k}$ shall not overflow and shall be empty at least once every second;
- Each $MB_{l,k}$, EB_l, and DPB shall not overflow;
- EB_l shall not underflow, except when VUI parameters are present for the HEVC video sequence with the low_delay_hrd_flag set to '1'. Underflow of EB_l occurs for HEVC access unit $A_n(j)$ when one or more bytes of $A_n(j)$ are not present in EB_l at the decoding time $td_n(j)$.

2.18 Carriage of green access units

2.18.1 Carriage of green access units in MPEG-2 sections

Green access units are carried using the MPEG-2 private section syntax with the section_syntax_indicator element set to '0'. See Table 2-138.

Table 2-138 – Green access unit section syntax

Syntax	Bits	Mnemonic
Green_access_unit_section_message(){		
table_ID	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
private_indicator	**1**	**bslbf**
reserved	**2**	**bslbf**
private_section_length	**12**	**uimsbf**
'0010'	**4**	**bslbf**
Display_in_PTS [32..30]	**3**	**bslbf**
marker_bit	**1**	**bslbf**
Display_in_PTS [29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
Display_in_PTS [14..0]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
Green_Au()		
CRC_32	**32**	**rpchof**
}		

2.18.2 Semantics of green access unit section

table_id – This shall be set to 0x09.

section_syntax_indicator – This shall be set to '0'.

Display_in_PTS – This is the 33-bit PTS specified similar to that defined in the PES header and is used with the associated video access unit.

Green_Au() – Defined in 2.18.3.

2.18.3 Green access unit

The format of the green access unit is defined in Table 2-139. Green access units contain dynamic metadata and are carried in MPEG private section format.

Table 2-139 – Green access unit

Syntax	No. bits	Mnemonic
Green_Au {		
num_quality_levels	**4**	**uimsbf**
reserved	**4**	**bslbf**
for (k=0; k < num_constant_backlight_voltage_time_intervals; k++) {		
for (j=0; j < num_max_variations; j++) {		
lower_bound	**8**	**uimsbf**
if (lower_bound > 0)		
upper_bound	**8**	**uimsbf**
rgb_component_for_infinite_psnr	**8**	**uimsbf**
for (i=1; i <= num_quality_levels; i++){		
max_rgb_component	**8**	**uimsbf**
scaled_psnr_rgb	**8**	**uimsbf**
}		
}		
}		
}		

As explained in 6.4 of ISO/IEC 23001-11, each combination of constant_backlight_voltage_time_interval and max_variation is associated with contrast-enhancement metadata and a set of quality levels defined in Table 2-139.

The metadata in the Green_AU is applicable to the presentation subsystem until the next Green_AU containing metadata arrives.

Semantics for all the elements in Table 2-139 is defined in 6.4 of ISO/IEC 23001-11.

2.18.4 Timing relationship between green access unit and media access unit

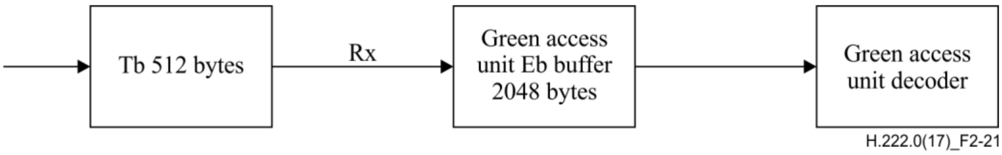
The green access unit should be decoded and information should be available before the associated media access unit is decoded. Such a timing relationship guarantees that the metadata within the green access unit is made available to the display with sufficient lead time relative to the PTS of the associated media access unit. Note that the PTS of the media access unit and the PTS of the green access unit are identical. The green access unit is transmitted in the transport stream with a sufficient lead time so that the display control settings can be adjusted in advance of presentation time for correct operation. If num_constant_backlight_voltage_time_intervals > 1, then the lead time should be equal to or larger than the largest constant_backlight_voltage_time_interval. The PMT shall not contain more than one green metadata component (stream_type equal to 0x2C).

NOTE – Applications that use carouseling of green access unit data carouseling in a given program can do so as long as the display_in_PTS value is adjusted to conform to the PCR clock and T-STD buffer.

2.18.5 Buffer model for processing green access units

The buffer model reflects the processing required to handle green access units. The model can be used to establish constraints which can be used to verify the validity of dynamic green metadata prepared in accordance with this standard.

Figure 2-21 illustrates the buffer model for processing green access units.



MPEG-2 transport stream packets come into the model at the left, and are filtered by PID. Packets whose PID matches the green access unit PID flow into the 512 byte transport buffer. These buffered packets are removed at a rate of Rx = 300 kbps (kilobits/second) and stored in the green access unit Eb buffer (2048 Bytes). Green access unit table sections are removed from the Eb buffer immediately after the full access unit is available (based on section length) and these are passed onto the green access unit decoder at a rate Rbx = 300 kbps for decoding and each decoded access unit is associated with the video at time = display_in_PTS. The Eb buffer shall not overflow and the green access unit section shall be available in the Eb buffer at least 100 ms before display_in_PTS.

NOTE – In the worst-case, a green AU would contain 4488 bits. Under such conditions, the Eb buffer is large enough to hold up to three green AUs and the rate Rx is high enough to allow the removal of green AUs that are associated with video frames at 60 fps.

2.19 Carriage of ISO/IEC 23008-3 MPEG-H 3D audio data

2.19.1 Introduction

An ITU-T Rec. H.222.0 | ISO/IEC 13818-1 stream may carry ISO/IEC 23008-3 audio elementary streams. Typically, an ISO/IEC 23008-3 stream will be an element of an ITU-T Rec. H.222.0 | ISO/IEC 13818-1 program, as defined by the PMT in a Transport Stream and the PSM in a Program Stream.

2.19.2 Carriage in PES packets

Individual ISO/IEC 23008-3 elementary streams shall be carried in PES packets as PES_packet_data_bytes. For PES packetization no specific data alignment constraints apply. For synchronization PTS is encoded in the header of the PES packet that carries the ISO/IEC 23008-3 elementary stream data; for PTS encoding the same constraints apply as for ISO/IEC 13818 elementary streams.

Before PES packetization the elementary stream data shall be first encapsulated in the MHAS transport syntax defined in clause 14 of ISO/IEC 23008-3. If a PTS is present in the PES packet header, it shall refer to the first mpeghAudioStreamPacket (MHAS packet) of MHASPacketType PACTYP_MPEGH3DAFRAME that commences in the payload of the PES packet. This MHAS packet shall occur after the first MHAS packet of MHASPacketTypes PACTYP_SYNC and PACTYPE_MPEGH3DACFG that commences in the payload of the PES packet, if any are present. Note that additional MHAS packets of other packet types may be present between those MHAS packets (PACTYP_SYNC, PACTYPE_MPEGH3DACFG and PACTYP_MPEGH3DAFRAME), e.g., PACTYP_SYNCGAP, PACTYP_AUDIOSCENEINFO or PACTYP_AUDIOTRUNCATION.

It is recommended but not mandatory that in this case the first byte of the PES_packet_data_bytes is the first byte of an mpeghAudioStreamPacket with MHASPacketType set to PACTYP_SYNC. In this case, the data_alignment_indicator in the PES packet header is set to '1'. According to 14.4.4 of ISO/IEC 23008-3, the mpeghAudioStreamPacket yields an overall syncword of 0xC001A5.

To provide appropriate specific information, it is strongly recommended that each MPEG-H 3D audio stream carries audio scene information data with sufficient information to ensure that the decoded and rendered MPEG-H 3D audio stream can be presented correctly by receivers.

Carriage of ISO/IEC 23008-3 elementary streams in PES packets shall be identified by appropriate stream_id and stream_type values, indicating the use of ISO/IEC 23008-3 3D audio. In addition, an MPEG-H_3dAudio_descriptor() shall be present in the descriptor loop for the respective elementary stream entry in the Program Map Table in case of a transport stream.

2.19.3 STD extensions for ISO/IEC 23008-3 elementary streams

The T-STD model as defined in 2.4.2.1 includes a transport buffer TB_n and a multiplex buffer B_n prior to decoding of each ISO/IEC 23008-3 3D audio elementary stream n. For buffers TB_n and B_n and the rate Rx_n between TB_n and B_n the following constraints apply for the carriage of an ISO/IEC 23008-3 3D audio stream:

The size BS_n of the buffer B_n, is defined as $BS_n = BS_{mux} + BS_{dec} + BS_{oh}$ with:

BS_n = 3584 bytes for 1-2 encoded audio channel signals

Here, the size of the access unit decoding buffer BS_{dec}, and the PES packet overhead buffer BS_{oh} are constrained by: $BS_{dec} + BS_{oh} \leq 2848$ bytes; a portion (736 bytes) of the 3584 byte buffer is allocated for buffering to allow multiplexing. The rest, 2848 bytes, are shared for access unit buffering BS_{dec}, BS_{oh} and additional multiplexing.

BS_n = 8976 bytes for 3-8 encoded audio channel signals

BS_n = 12804 bytes for 9-12 encoded audio channel signals

BS_n = 51216 bytes for 13-48 encoded audio channel signals

BS_n = 136576 bytes for 49-128 encoded audio channel signals

Rate Rx_n:

Rx_n = 2 000 000 bit/s for 1-2 encoded audio channel signals

Rx_n = 5 529 600 bit/s for 3-8 encoded audio channel signals

Rx_n = 8 294 400 bit/s for 9-12 encoded audio channel signals

Rx_n = 33 177 600 bit/s for 13-48 encoded audio channel signals

Rx_n = 88 473 600 bit/s for 49-128 encoded audio channel signals

2.19.4 STD extensions for multiple ISO/IEC 23008-3 elementary streams

ISO/IEC 23008-3 MPEG-H 3D audio allows encoding an audio program as several elementary streams. One single audio decoder decodes all elementary streams to one audio presentation. Each of those elementary streams carries one or more encoded audio channel signals.

For every single elementary stream the constraints for the buffer size BS_n and the rate Rx_n apply as described in 2.19.3 above.

As described in 14.6 of ISO/IEC 23008-3, the MHAS packets of each elementary stream are merged into one single stream before forwarding the data to the decoder. Note that based on user interactivity only a subset of the elementary streams may be merged into this single stream instead of the complete set of elementary streams.

Figure 2-22 illustrates an example of multiple elementary streams. Those streams are merged into one single intermediate stream and decoded by a single ISO/IEC 23008-3 3D audio decoder.

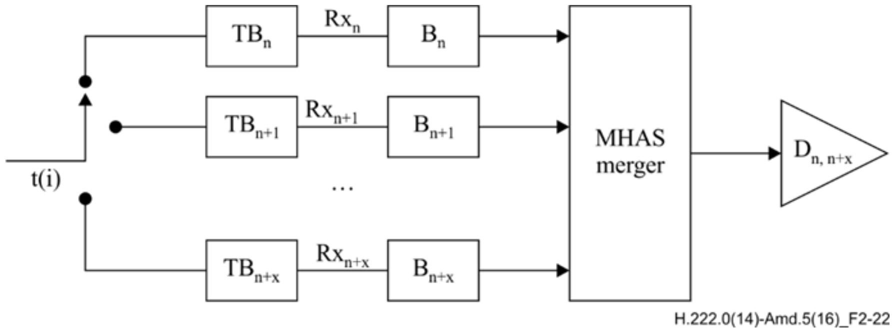


Figure 2-22 – Transport stream system target decoder for multiple audio elementary streams

In addition to the notation used and described in 2.4.2.1 to describe Figure 2-1 the following notation is used in Figure 2-22:

$D_{n,n+x}$ is the audio decoder for all x audio elementary streams n to n+x.

2.19.5 MPEG-2 Transport stream random access constraints and signalling

A TS packet containing the PES packet header of a 3D audio random access point (RAP) shall have an adaptation field. The payload_unit_start_indicator bit shall be set to '1' in the TS packet header and the adaptation_field_control bits shall be set to '11' (as per section 2.4.3.3). In addition, the random_access_indicator bit in the adaptation field of the TS packet that contains the PES packet header of the 3D audio RAP shall be set to '1' and follow the constraints specified in 2.4.3.5.

If the PES packet contains a RAP, then the MHAS packet of type PACTYP_MPEGH3DAFRAME containing the RAP shall be the first MHAS packet of that type that commences in the PES packet. The audio data encapsulated in this MHAS packet shall follow the rules for a random access point as defined in 5.7 of ISO/IEC 23008-3.

2.20 Carriage of Quality Access Units in MPEG-2 sections

2.20.1 General description

Quality Access Units carrying dynamic quality metadata associated with one or more video frames, i.e., a given quality metric sample can be applicable for multiple video frames for a given DTS interval, until a new quality metric sample is declared. These access units are encapsulated in MPEG sections identified by stream_type value of 0x2F.

Each Quality Access Unit contains configuration and timing information, as well as an array of quality values, corresponding one for one to the declared metrics. Each value is padded by preceding zero bytes, as needed, to the number of bytes indicated by field_size_bytes.

Quality_access_unit shall be carried in MPEG sections.

Every Quality Access Unit shall be a random access point.

NOTE – in order to make processing easier, each Quality Access Unit should be contained in a single TS packet.

The syntax for the Quality Access Units is provided in Table 2-140. The attribute definitions are aligned with those in ISO/IEC 23001-10.

Table 2-140 – Quality Access Unit

Syntax	No. bits	Mnemonic
Quality_Access_Unit() {		
field_size_bytes	**8**	**uimsbf**
metric_count	**8**	**uimsbf**
for (i = 0; i < metric_count; i++) {		
metric_code	**32**	**uimsbf**
sample_count	**8**	**uimsbf**
for (j = 0; j < sample_count; j++) {		
'0010'	**4**	**bslbf**
media_DTS [32..30]	**3**	**uimsbf**

Syntax	No. bits	Mnemonic
marker_bit	**1**	**bslbf**
media_DTS [29..15]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
media_DTS [14..0]	**15**	**uimsbf**
marker_bit	**1**	**bslbf**
quality_metric_sample	**N**	**uimsbf**
}		
}		
}		

field_size_bytes – Size of the quality_metric_sample field. As such, N=8*field_size_bytes

metric_count – The number of metrics for quality values in the Quality Access Unit, as defined in 4.2 of ISO/IEC 23001-10.

metric_code – The code name of the metrics in the Quality Access unit, as defined in 4.2 of ISO/IEC 23001-10.

sample_count – Number of quality metric samples per metric.

media_DTS – DTS of the media access unit described by the quality metric sample.

As such, quality_metric_sample is an array signalled using the 'i' and 'j' values of the loop.

2.20.2 Buffer model for processing Quality Access Units

The buffer model reflects processing required to handle Quality Access Units. The model can be used to establish constraints which can be used to verify the validity of dynamic Quality metadata prepared in accordance with this Standard.

Figure 2-23 illustrates the buffer model for processing Quality Access Units.

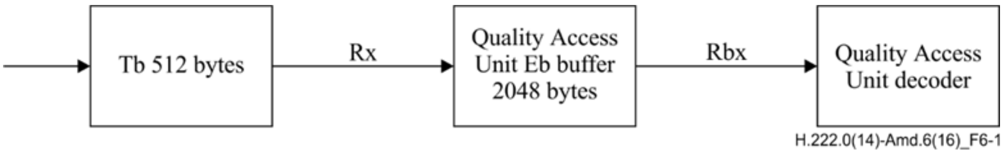


Figure 2-23 – Quality Access Unit decoder processing model

MPEG-2 Transport Stream packets come into the model at the left, and are filtered by PID. Packets whose PID matches the Quality Access Unit PID flow into the 512 byte transport buffer. These buffered packets are removed at a rate of Rx = 300 kbps (kilobits/second) and stored in the Quality Access Unit Eb buffer (2048 Bytes). Quality Access Unit table sections are removed from the Eb buffer immediately after the full access unit is available (based on section length) and these are passed onto the Quality Access Unit decoder at a rate Rbx = 300 kbps for decoding and association with the corresponding video frames. Eb buffer shall not overflow and the Quality Access Unit section shall be available in the Eb buffer to meet the requirement of DTS earlier than the DTS of the latest media access unit described in the corresponding Quality Access Unit.

Annex A

CRC decoder model

(This annex forms an integral part of this Recommendation | International Standard.)

A.1 CRC decoder model

The 32-bit CRC decoder model is specified in Figure A.1.

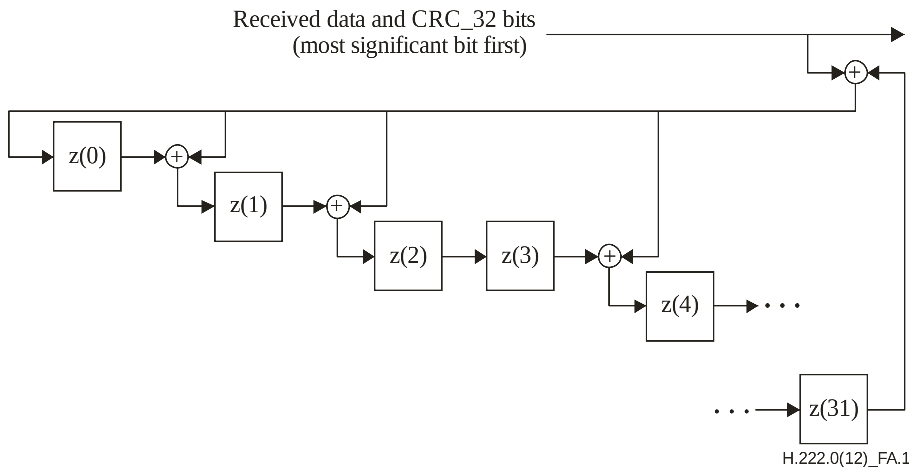


Figure A.1 – 32-bit CRC decoder model

The 32-bit CRC decoder operates at bit level and consists of 14 adders '+' and 32 delay elements z(i). The input of the CRC decoder is added to the output of z(31), and the result is provided to the input z(0) and to one of the inputs of each remaining adder. The other input of each remaining adder is the output of z(i), while the output of each remaining adder is connected to the input of z(i + 1), with i = 0, 1, 3, 4, 6, 7, 9, 10, 11, 15, 21, 22, and 25. Refer to Figure A.1.

This is the CRC calculated with the polynomial:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1 \quad \text{(A-1)}$$

Bytes are received at the input of the CRC decoder. Each byte is shifted into the CRC decoder one bit at a time, with the left most bit (msb) first. For example, if the input is byte 0x01 the seven '0's enter the CRC decoder first, followed by the one '1'. Before the CRC processing of the data of a section the output of each delay element z(i) is set to its initial value '1'. After this initialization, each byte of the section is provided to the input of the CRC decoder, including the four CRC_32 bytes. After shifting the last bit of the last CRC_32 byte into the decoder, i.e., into z(0) after the addition with the output of z(31), the output of all delay elements z(i) is read. In the case where there are no errors, each of the outputs of z(i) shall be zero. At the CRC encoder the CRC_32 field is encoded with a value such that this is ensured.

Annex B

Digital storage medium command and control (DSM-CC)

(This annex does not form an integral part of this Recommendation | International Standard.)

B.1 Introduction

The DSM-CC protocol is a specific application protocol intended to provide the basic control functions and operations specific to managing a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream on digital storage media. This DSM-CC is a low-level protocol above network/OS layers and below application layers.

The DSM-CC shall be transparent in the following sense:

- It is independent of the DSM used;
- it is independent of whether the DSM is located at a local or remote site;
- it is independent of the network protocol with which the DSM-CC is interfaced;
- it is independent of the various operating systems on which the DSM is operated.

B.1.1 Purpose

Many applications of Rec. ITU-T H.222.0 | ISO/IEC 13818-1 DSM Control Commands require access to a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream stored on a variety of digital storage media at a local or remote site. Different DSM have their own specific control commands and thus, a user would need to know different sets of specific DSM control commands in order to access Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstreams from different DSM. This brings many difficulties to the interface design of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 or ISO/IEC 11172-1 application system. To overcome this difficulty, a set of common DSM control commands, which is independent of the specific DSM used, is suggested in this annex. This annex is informative only. ISO/IEC 13818-6 defines DSM-CC extension with a broader scope.

B.1.2 Future applications

Beyond the immediate applications supported by the current DSM control commands, future applications based on extensions of DSM command control could include the following:

Video on demand

Video programs are provided as requested by a customer through various communication channels. The customer could select a video program from a list of programs available from a video server. Such applications could be used by hotels, cable TV, educational institutions, hospitals, etc.

Interactive video services

In these applications, the user provides frequent feedback controlling the manipulation of stored video and audio. These services can include video-based games, user-controlled video tours, electronic shopping, etc.

Video networks

Various applications may wish to exchange stored audio and video data through some type of computer network. Users could route AV information through the video network to their terminals. Electronic publishing and multimedia applications are examples of this kind of application.

B.1.3 Benefits

Specifying the DSM control commands independent of the DSM, end-users can perform Rec. ITU-T H.222.0 | ISO/IEC 13818-1 decoding without having to fully understand the detailed operation of the specific DSM used.

The DSM control commands are codes to give end users the assurance that the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstreams can be played and stored with the same semantics, independent of the DSM and user interface. They are fundamental commands for the control of DSM operation.

B.1.4 Basic functions

B.1.4.1 Stream selection

The DSM-CC provides the means to select a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream upon which to perform the succeeding operations. Such operations include creation of a new bitstream. Parameters of this function include:

- index of the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream (the mapping between this index and a name meaningful to an application is outside the scope of the current DSM-CC);
- mode (retrieval/storage).

B.1.4.2 Retrieval

The DSM-CC provides the means to:

- play an identified Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream;
- play from a given presentation time;
- set the playback speed (normal or fast);
- set the playback duration (until a specified presentation time, the end of the bitstream in forward play or the beginning in reverse play or the issuance of a stop command);
- set the direction (forward or reverse);
- pause;
- resume;
- change the access point in the bitstream;
- stop.

B.1.4.3 Storage

The DSM-CC provides the means to:

- cause storage of a valid bitstream for a specified duration;
- cause storage to stop.

DSM-CC provides a useful but limited subset of functionality that may be required in DSM based Rec. ITU-T H.222.0 | ISO/IEC 13818-1 applications. It is fully expected that significant additional capabilities will be added through subsequent extensions.

B.2 General elements

B.2.1 Scope

The scope of this work consists of the development of a Recommendation | International Standard to specify a useful set of commands for control of digital storage media on which a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream is stored. The commands can perform remote control of a digital storage media in a general way independently of the specific DSM and apply to any Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream stored on a DSM.

B.2.2 Overview of the DSM-CC application

The current DSM-CC syntax and semantics cover the single user to DSM application. The user's system is capable of retrieving a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream and is also (optionally) capable of generating a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream. The control channel over which the DSM commands and acknowledgements are sent is shown in Figure B.1 as an out-of-band channel. This can also be accomplished by inserting the DSM-CC commands and acknowledgements into the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstreams if an out-of-band channel is not available.

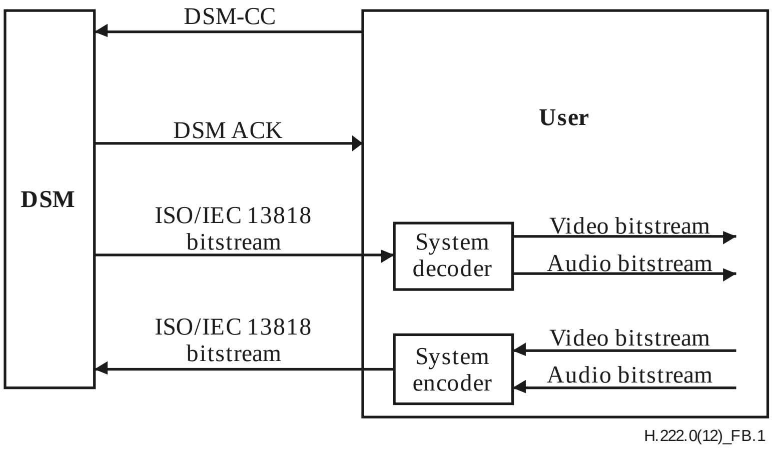


Figure B.1 – Configuration of DSM-CC application

B.2.3 The transmission of DSM-CC commands and acknowledgements

The DSM-CC is encoded into a DSM-CC bitstream according to the syntax and semantics defined in B.3.2 through B.3.9. The DSM-CC bitstream can be transmitted both as a stand-alone bitstream and in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Systems bitstream.

When the DSM-CC bitstream is transmitted in stand-alone mode, its relationship to the Systems bitstream and the decoding process is illustrated in Figure B.2. In this case, the DSM-CC bitstream is not embedded in the Systems bitstream. This transmission mode can be used in the applications when the DSM is connected directly with the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 decoder. It can also be used in the applications where the DSM-CC bitstream could be controlled and transmitted by other types of network multiplexors.

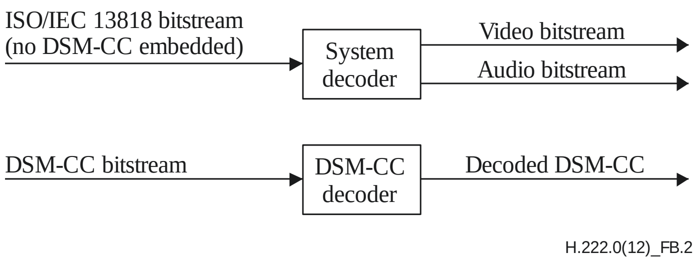


Figure B.2 – BSM-CC bitstream decoded as a stand-alone bitstream

For some applications, it is desirable to transmit the DSM-CC in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 systems bitstream so that some features of the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 systems bitstream could be applied to the DSM-CC bitstream as well. In this case, the DSM-CC bitstream is embedded in the systems bitstream by the systems multiplexor.

The DSM-CC bitstream is encoded by the systems encoder in the following process. First, the DSM-CC bitstream is packetized into a packetized elementary stream (PES) according to the syntax described in 2.4.3.6. The PES packet is then multiplexed into either a program stream (PS) or a transport stream (TS) according to the requirement of the transmission media. The decoding procedures are the inverse of the encoding procedures and are illustrated in the block diagram of the Systems decoder depicted in Figure B.3.

In Figure B.3, the output of the Systems decoder is a video bitstream, audio bitstream and/or DSM-CC bitstream. The DSM-CC bitstream is identified by the stream_id, value '1111 0010' as defined by the stream_id Table 2-22. Once the DSM-CC bitstream is identified, it follows the rules as specified by T-STD or P-STD.

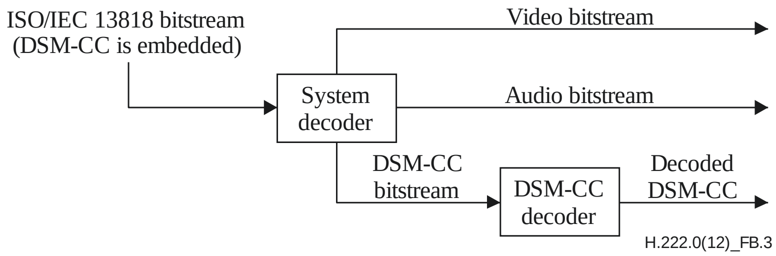


Figure B.3 – DSM-CC bitstream decoded as part of the system bitstream

B.3 Technical elements

B.3.1 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply:

B.3.1.1 DSM-CC: Digital Storage Media Command and Control Commands that are specified by Rec. ITU-T H.222.0 | ISO/IEC 13818-1 for the control of digital storage media at a local or remote site containing a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream.

B.3.1.2 DSM ACK: The acknowledgement from the DSM-CC command receiver to the command initiator.

B.3.1.3 MPEG bitstream: An ISO/IEC 11172-1 Systems stream, Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program stream or Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport stream.

B.3.1.4 DSM-CC server: A system, either local or remote, used to store and/or retrieve a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream.

B.3.1.5 point of random access: A point in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream with the property that for at least one elementary stream within the bitstream, the next access unit, 'N', completely contained in the bitstream can be decoded without reference to previous access units, and for every elementary stream in the bitstream all access units with the same or later presentation times are completely contained subsequently in the bitstream and can be completely decoded by a system target decoder without access to information prior to the point of random access. The bitstream as stored on the DSM may have certain points of random access; the output of the DSM may include additional points of random access manufactured by the DSM's own manipulation of the stored material (e.g., storing quantization matrices so that a sequence header can be generated whenever necessary). A point of random access has an associated PTS, namely the actual or implied PTS of access unit 'N'.

B.3.1.6 current operational PTS value: The actual or implied PTS associated with the last point of random access preceding the last access unit provided from the DSM from the currently selected Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream. If no access unit has been provided from this Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream, the DSM is incapable of providing random access into the current bitstream, then the current operational PTS value is the first point of random access in the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream.

B.3.1.7 DSM-CC bitstream: A sequence of bits satisfying the syntax of B.3.2.

B.3.2 Specification of DSM-CC syntax

- Every DSM control command shall commence with a start_code, as specified in Table B.1.
- Every DSM control command shall have a packet_length to specify the number of byte in a DSM-CC packet.
- When the DSM-CC bitstream is transmitted as a PES packet as defined in 2.4.3.6, the fields up to the packet_length field are identical to those specified in 2.4.3.6. In other words, if the DSM-CC packet is encapsulated in a PES packet, the PES packet start code is the only start code at the beginning of the packet.
- The actual control command or acknowledgement shall follow the last byte of the packet_length field.
- An acknowledgement stream shall be provided by the DSM control bitstream receiver after the requested operation is started or is completed, depending on the command received.
- At all times the DSM is responsible for providing a normative Rec. ITU-T H.222.0 | ISO/IEC 13818-1 stream. This may include manipulating the trick mode bits defined in 2.4.3.6.

Table B.1 – DSM-CC syntax

Syntax	No. of bits	Mnemonic
DSM_CC() {		
packet_start_code_prefix	**24**	**bslbf**
stream_id	**8**	**uimsbf**
packet_length	**16**	**uimsbf**
command_id	**8**	**uimsbf**
If (command_id = = '01') {		
control()		
} else if (command_id = = '02') {		
ack()		
}		
}		

B.3.3 Semantics of fields in specification of DSM-CC syntax

packet_start_code_prefix – This is a 24-bit code. Together with the stream_id that follows it constitutes a DSM-CC packet start code that identifies the beginning of a DSM-CC packet bitstream. The packet_start_code_prefix is the bit string '0000 0000 0000 0000 0000 0001' (0x000001).

stream_id – This 8-bit field specifies the bitstream type and shall have a value '1111 0010' for the DSM-CC bitstream. Refer to Table 2-22.

packet_length – This 16-bit field specifies the number of bytes in the DSM-CC packet immediately following the last byte of this field.

command_id – This 8-bit unsigned integer identifies the bitstream is a control command or an acknowledgement stream. The values are defined in Table B.2.

Table B.2 – Command_id assigned values

Value	Command_id
0x00	Forbidden
0x01	Control
0x02	Ack
0x03 .. 0xFF	Reserved

B.3.4 Control layer

Constraints on setting flags in DSM-CC control

- At most one of the flags for select, playback and storage shall be set to '1' for each DSM control command. If none of these bits are set, then this command shall be ignored.
- At most one of pause_mode, resume_mode, stop_mode, play_flag, and jump_flag shall be set for each retrieval command. If none of these bits are set, then this command shall be ignored.
- At most one of record_flag and stop_mode shall be selected for each storage command. If none of these bits are set, then this command shall be ignored.

See Table B.3.

Table B.3 – DSM-CC control

Syntax	No. of bits	Mnemonic
control() {		
select_flag	**1**	**bslbf**
retrieval_flag	**1**	**bslbf**
storage_flag	**1**	**bslbf**
reserved	**12**	**bslbf**
marker_bit	**1**	**bslbf**
If (select_flag = = '1') {		
bitstream_id [31..17]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
bitstream_id [16..2]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
bitstream_id [1..0]	**2**	**bslbf**
select_mode	**5**	**bslbf**
marker_bit	**1**	**bslbf**
}		
if (retrieve_flag = = '1') {		
jump_flag	**1**	**bslbf**
play_flag	**1**	**bslbf**
pause_mode	**1**	**bslbf**
resume_mode	**1**	**bslbf**
stop_mode	**1**	**bslbf**
reserved	**10**	**bslbf**
marker_bit	**1**	**bslbf**
if (jump_flag = ='1') {		
reserved	**7**	**bslbf**
direction_indicator	**1**	**bslbf**
time_code()		
}		
if (play_flag = ='1'){		
speed_mode	**1**	**bslbf**
direction_indicator	**1**	**bslbf**
reserved	**6**	**bslbf**
time_code()		
}		
}		
if (storage_flag = ='1') {		
reserved	**6**	**bslbf**
record_flag	**1**	**bslbf**
stop_mode	**1**	**bslbf**
if (record_flag = ='1') {		
time_code()		
}		
}		
}		

B.3.5 Semantics of fields in control layer

marker_bit – This is a 1-bit marker that is always set to '1' to avoid start code emulation.

reserved_bits – This 12-bit field is reserved for future use by this Recommendation | International Standard for DSM control commands. Until otherwise specified by ITU-T | ISO/IEC it shall have the value '0000 0000 0000'.

select_flag – This 1-bit flag when set to '1' specifies a bitstream selection operation. When it is set to '0' no bitstream selection operation shall occur.

retrieval_flag – This 1-bit flag when set to '1' specifies that a specific retrieval (playback) action will occur. The operation starts from the current operational PTS value.

storage_flag – This 1-bit flag when set to '1' specifies that a storage operation is to be executed.

bitstream_ID – This 32-bit field is coded in three parts. The parts are combined to form an unsigned integer specifying which Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream is to be selected. It is the DSM server's responsibility to map the names of the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstreams stored on its DSM uniquely to a series of numbers which could be represented by the bitstream_ID.

select_mode – This 5-bit unsigned integer specifies which mode of bitstream operation is requested. Table B.4 specifies the defined modes.

Table B.4 – Select mode assigned values

Code	Mode
0x00	Forbidden
0x01	Storage
0x02	Retrieval
0x03 .. 0x1F	Reserved

jump_flag – This 1-bit flag when set to '1' specifies a jump in the playback pointer to a new access unit. The new PTS is specified by a relative time_code with respect to the current operational PTS value. This function is only valid when the current Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream is in the "stop" mode.

play_flag – This 1-bit flag when set to '1' specifies to play a bitstream for a certain time period. The speed, direction, and play duration are additional parameters in the bit stream. The play starts from the current operational PTS value.

pause_mode – This is a one-bit code specifying to pause the playback action and keep the playback pointer at the current operational PTS value.

resume_mode – This is a one-bit code specifying to continue the playback action from the current operational PTS value. Resume only has meaning if the current bitstream is in the "pause" state, and the bitstream will be set to the forward play state at normal speed.

stop_mode – This is a one-bit code specifying to stop a bitstream transmission.

direction_indicator – This is a one-bit code to indicate the playback direction. If this bit is set to '1', it stands for a forward play. Otherwise it stands for a backward play.

speed_mode – This is a 1-bit code to specify the speed scale. If this bit is set to '1', it specifies that the speed is normal play. If this bit is set to '0', it specifies that the speed is fast play (i.e., fast forward or fast reverse).

record_flag – This is one-bit flag to specify the request of recording the bitstream from an end user to a DSM for a specified duration or until the reception of a stop command, whichever comes first.

B.3.6 Acknowledgement layer

Constraints on setting flags in DSM-CC control

Only one of the acks bits specified below can be set to '1' for each DSM ack bitstream (see Table B.5).

Table B.5 – DSM-CC Acknowledgement

Syntax	No. of bits	Mnemonic
ack() {		
select_ack	**1**	**bslbf**
retrieval_ack	**1**	**bslbf**
storage_ack	**1**	**bslbf**
error_ack	**1**	**bslbf**
reserved	**10**	**bslbf**
marker_bit	**1**	**bslbf**
cmd_status	**1**	**bslbf**
If (cmd_status = = '1' &&		
(retrieval_ack = ='1' || storage_ack = = '1')) {		
time_code()		
}		
}		

B.3.7 Semantics of fields in acknowledgement layer

select_ack – This 1-bit field when it is set to '1' indicates that the ack() command is to acknowledge a select command.

retrieval_ack – This 1-bit field when set to '1' indicates that the ack() command is to acknowledge a retrieval command.

storage_ack – This 1-bit field when set to '1' indicates that the ack() command is to acknowledge a storage command.

error_ack – This 1-bit field when set to '1' indicates a DSM error. The defined errors are EOF (end of file on forward play or start of file on reverse play) on a stream being retrieved and Disk Full on a stream being stored. If this bit is set to '1', cmd_status is undefined. The current bitstream is still selected.

cmd_status – This 1-bit flag set to '1' indicates that the command is accepted. When set to '0' it indicates the command is rejected. The semantics vary according to the command received as follows:

- If select_ack is set and cmd_status is set to '1', it specifies that the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream is selected and the server is ready to provide the selected mode of operation. The current operational PTS value is set to the first point of random access of the newly selected Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream. If cmd_status is set to '0', the operation has failed and no bitstream is selected.
- If retrieval_ack is set and cmd_status is set to '1', it specifies that the retrieval operation is initiated for all retrieval commands. The position of the current operational PTS pointer is reported by the succeeding time_code.
- For the play_flag command with infinite_time_flag != '1', a second acknowledgement will be sent. This will acknowledge that the play operation has ended by reaching the duration defined by the play_flag command.
- If the cmd_status is set to '0' in a retrieval acknowledgement, the operation has failed. Possible reasons for this failure include an invalid bitstream_ID, jumping beyond the end of a file, or a function not supported such as reverse play in standard speed.
- If storage_ack is set, it specifies that the storage operation is being started for the record_flag command or is completed by the stop_mode command. The PTS of the last complete access unit stored is reported by the succeeding time_code.
- If the recording operation is ended by reaching the duration defined by the storage_flag command, another acknowledgement shall be sent and the current operational PTS value after the recording shall be reported.
- If the cmd_status is set to '0' in a storage acknowledgement, the operation has failed. Possible reasons for this failure include an invalid bitstream_ID, or the inability of the DSM to store data.

B.3.8 Time code

Constraints on time code

- A forward operation of specified duration given by a time_code terminates after the actual or implied PTS of an access unit is observed such that PTS minus the current operational PTS value at the start of the operation modulo 2^{33} exceeds the duration.
- A backward operation of specified duration given by a time_code terminates after the actual or implied PTS of an access unit is observed such that current operational PTS value at the start of the operation minus that PTS modulo 2^{33} exceeds the duration.
- For all the commands in the control() layer, the time_code is specified as a relative duration with respect to the current operational PTS value.
- For all the commands in the ack() layer, the time_code is specified by the current operational PTS value.

See Table B.6.

Table B.6 – Time code

Syntax	No. of bits	Mnemonic
time_code() {		
reserved	**7**	**bslbf**
infinite_time_flag	**1**	**bslbf**
if (infinite_time_flag = = '0') {		
reserved	**4**	**bslbf**
PTS [32..30]	**3**	**bslbf**
marker	**1**	**bslbf**
PTS [29..15]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
PTS [14..0]	**15**	**bslbf**
marker_bit	**1**	**bslbf**
}		
}		

B.3.9 Semantics of fields in time code

infinite_time_flag – This 1-bit flag when set to '1' indicates an infinite time period. This flag is set to '1' in applications where a time period for a specific operation could not be defined in advance.

PTS [32..0] – The presentation timestamp of the access unit of the bitstream. Depending upon the function, this can be an absolute value or a relative time delay in cycles of the 90-kHz system clock.

Annex C

Program-specific information

(This annex does not form an integral part of this Recommendation | International Standard.)

C.1 Explanation of program-specific information in transport streams

Clause 2.4.4 contains the normative syntax, semantics and text concerning program-specific information. In all cases, compliance with the constraints of 2.4.4 is required. This annex provides explanatory information on how to use the PSI functions, and considers examples of how it may be used in practice.

C.2 Introduction

This Recommendation | International Standard provides a method for describing the contents of transport stream packets for the purpose of the demultiplexing and presentation of programs. The coding specification accommodates this function through the program-specific information (PSI). This annex discusses the use of PSI.

The PSI may be thought of as belonging to six tables:

1) Program association table (PAT);
2) TS program map table (PMT);
3) Network information table (NIT);
4) Conditional access table (CAT).
5) Transport stream description table (TSDT); and
6) IPMP control information table.

The contents of the PAT, PMT, CAT and TSDT are specified in this Recommendation | International Standard. ICIT is defined in ISO/IEC 13818-11 (MPEG-2 IPMP).

The NIT is a private table, and the PID value of the transport stream packets which carry it is specified in the PAT. Both the NIT and ICIT must follow the structure defined in this Recommendation | International Standard.

C.3 Functional mechanism

The tables listed above are conceptual in that they need never be regenerated in a specified form within a decoder. While these structures may be thought of as simple tables, they may be partitioned before they are sent in transport stream packets. The syntax supports this operation by allowing the tables to be partitioned into sections and by providing a normative mapping method into transport stream packet payloads. A method is also provided to carry private data in a similar format. This is advantageous as the same basic processing in the decoder can then be used for both the PSI data and the private data helping to keep cost down. For advice on the optimum placing of PSI in the transport stream, see Annex D.

Each section is uniquely identified by the combination of the following elements:

i) **table_id**

   The 8-bit table_id identifies to which table the section belongs.

   - Sections with table_id 0x00 belong to the Program Association Table.
   - Sections with table_id 0x01 belong to the Conditional Access Table.
   - Sections with table_id 0x02 belong to the TS Program Map Table.
   - Sections with table_id 0x03 belong to the TS_description_section.
   - Sections with table_id 0x04 belong to the ISO_IEC_14496_scene_description_section.
   - Sections with table_id 0x05 belong to the ISO_IEC_14496_object_descriptor_section.
   - Sections with table_id 0x06 belong to the metadata_section.
   - Sections with table_id 0x07 belong to the IPMP_Control_Information_section.

   Other values of the table_id can be allocated by the user for private purposes.

   It is possible to set up filters looking at the table_id field to identify whether a new section belongs to a table of interest or not.

ii) **table_id_extension**

This 16-bit field exists in the long version of a section. In the Program Association Table it is used to identify the transport_stream_id of the stream – effectively a user-defined label which allows one transport stream to be distinguished from another within a network or across networks. In the conditional access table this field currently has no meaning and is therefore marked as "reserved" meaning that it shall be coded as 0xFFFF, but that a meaning may be defined by ITU-T | ISO/IEC in a subsequent revision of this Recommendation | International Standard. In a TS Program Map section the field contains the program_number, and thereby identifies the program to which the data in the section refers. The table_id_extension can also be used as a filter point in certain cases.

iii) **section_number**

The section_number field allows the sections of a particular table to be reassembled in their original order by the decoder. There is no obligation within this Recommendation | International Standard that sections must be transmitted in numerical order, but this is recommended, unless it is desired to transmit some sections of the table more frequently than others, e.g., due to random access considerations.

iv) **version_number**

When the characteristics of the transport stream described in the PSI change (e.g., extra programs added, different composition of elementary streams for a given program), then new PSI data has to be sent with the updated information as the most recently transmitted version of the sections marked as "current" must always be valid. Decoders need to be able to identify whether the most recently received section is identical with the section they have already processed/stored (in which case the section can be discarded), or whether it is different, and may therefore signify a configuration change. This is achieved by sending a section with the same table_id, table_id_extension, and section_number as the previous section containing the relevant data, but with the next value version_number.

v) **current_next_indicator**

It is important to know at what point in the bitstream the PSI is valid. Each section can therefore be numbered as valid "now" (current), or as valid in the immediate future (next). This allows the transmission of a future configuration in advance of the change, giving the decoder the opportunity to prepare for the change. There is however no obligation to transmit the next version of a section in advance, but if it is transmitted, then it shall be the next correct version of that section.

C.4 The mapping of sections into transport stream packets

Sections are mapped directly into transport stream packets, that is to say, without a prior mapping into PES packets. Sections do not have to start at the beginning of transport stream packets, (although they may), because the start of the first section in the payload of a transport stream packet is pointed to by the pointer_field. The presence of the pointer_field is signalled by the payload_unit_start_indicator being set to a value of '1' in PSI packets. (In non-PSI packets, the indicator signals that a PES packet starts in the transport stream packet.) The pointer_field points to the start of the first section in the transport stream packet. There is never more than one pointer_field in a transport stream packet, as the start of any other section can be identified by counting the length of the first and any subsequent sections, since no gaps between sections within a transport stream packet are allowed by the syntax.

It is important to note that within transport stream packets of any single PID value, one section must be finished before the next one is allowed to be started, or else it is not possible to identify to which section header the data belongs. If a section finishes before the end of a transport stream packet, but it is not convenient to open another section, a stuffing mechanism is provided to fill up the space. Stuffing is performed by filling each remaining byte of the packet with the value 0xFF. Consequently the table_id value 0xFF is forbidden, or else this would be confused with stuffing. Once a 0xFF byte has occurred at the end of a section, then the rest of the transport stream packet must be stuffed with 0xFF bytes, allowing a decoder to discard the rest of the transport stream packet. Stuffing can also be performed using the normal adaptation_field mechanism.

C.5 Repetition rates and random access

In systems where random access is a consideration, it is recommended to re-transmit PSI sections several times, even when changes do not occur in the configuration, as in the general case, a decoder needs the PSI data to identify the contents of the transport stream, to be able to start decoding. This Recommendation | International Standard does not place any requirements on the repetition or occurrence rate of PSI sections. Clearly though, repeating sections frequently helps random access applications, whilst causing an increase in the amount of bitrate used by PSI data. If program mappings are static or quasi-static, they may be stored in the decoder to allow faster access to the data than having to wait for it to be re-transmitted. The trade-off between the amount of storage required and the desired impact on channel acquisition time may be made by the decoder manufacturer.

C.6 What is a program?

The concept of a program has a precise definition within this Recommendation | International Standard [refer to 2.1.105 program (system)]. For a transport stream the time base is defined by the PCR. This effectively creates a virtual channel within the transport stream.

Note that this is not the same definition as is commonly used in broadcasting, where a "program" is a collection of elementary streams not only with a common timebase, but also with a common start and end time. A series of "broadcaster programs" (referred to in this annex as events) can be transmitted sequentially in a transport stream using the same program_number to create a "broadcasting conventional" TV-channel (sometimes called a service).

Event descriptions could be transmitted in private_sections().

A program is denoted by a program_number which has significance only within a transport stream. The program_number is a 16-bit unsigned integer and thus permits 65535 unique programs to exist within a transport stream (program_number 0 is reserved for identification of the NIT). Where several transport streams are available to the decoder (e.g., in a cable network), in order to successfully demultiplex a program, the decoder must be notified of both the transport_stream_id (to find the right multiplex) and the program_number of the service (to find the right program within the multiplex).

The transport stream mapping may be accomplished via the optional network information table. Note that the network information table may be stored in decoder non-volatile memory to reduce channel acquisition time. In this case, it needs to be transmitted only often enough to support timely decoder initialization set-up operations. The contents of the NIT are private, but shall take at least the minimum section structure.

C.7 Allocation of program_number

It may not be convenient in all cases to group together all the program elements which share a common clock reference as one program. It is conceivable to have a multi-service transport stream with only one set of PCRs, common to all. In general, though, a broadcaster may prefer to logically split up the transport stream into several programs, where the PCR_PID (location of the clock reference) is always the same. This method of splitting the program elements into pseudo-independent programs can have several uses. Two examples follow:

i) multilingual *transmissions into separate markets*

One video stream may be accompanied by several audio streams in different languages. It is advisable to include an example of the ISO_639_language_descriptor associated with each audio stream to enable the selection of the correct program and audio. It is reasonable to have several program definitions with different program_numbers, where all the programs reference the same video stream and PCR_PID, but have different audio PIDs. It is, however, also reasonable and possible to list the video stream and all the audio streams as one program, where this does not exceed the section size limit of 1024 bytes.

ii) *Very large program definitions*

There is a maximum limit on the length of a section of 1024 bytes (including section header and CRC_32). This means that no single program definition may exceed this length. For the great majority of cases, even with each program element having several descriptors, this size is adequate. However, one may envisage cases in very high bitrate systems, which could exceed this limit. It is then in general possible to identify methods of splitting the references of the streams, so that they do not all have to be listed together. Some program elements could be referenced under more than one program, and some under only one or the other, but not both.

C.8 Usage of PSI in a typical system

A communications system, especially in broadcast applications, may consist of many individual transport streams. Each one of the four PSI data structures may appear in each and every transport stream in a system. There must always be a complete version of the program association table listing all programs within the transport stream and a complete TS program map table, containing complete program definitions for all programs within the transport stream. If any streams are scrambled, then there must also be a conditional access table present listing the relevant Entitlement Management Messages (EMM) streams. The presence of a NIT is fully optional.

The PSI tables are mapped into transport stream packets via the section structure described above. Each section has a table_id field in its header, allowing sections from PSI tables and private data in private_sections to be mixed in transport stream packets of the same PID value or even in the same transport stream packet. Note, however, that within packets of the same PID, a complete section must be transmitted before the next section can be started. This is only possible for packets labelled as containing TS Program Map Table section or NIT packets however, since private sections may not be mapped into PAT or CAT packets.

It is required that all PAT sections be mapped into transport stream packets with PID = 0x0000 and all CA sections be mapped into packets with PID = 0x0001. PMT sections may be mapped into packets of user-selected PID value, listed as the PMT_PID for each program in the Program Association Table. Likewise, the PID for the NIT-bearing transport stream packets is user-selected, but must be pointed to by the entry "program_number = = 0x00" in the PAT, if the NIT exists.

The contents of any CA parameter streams are entirely private, but EMMs and ECMs must also be sent in transport stream packets to be compliant with this Recommendation | International Standard.

Private data tables may be sent using the private_section() syntax. Such tables could be used for example in a broadcasting environment to describe a service, an upcoming event, broadcast schedules and related information.

C.9 The relationships of PSI structures

Figure C.1 shows an example of the relationship between the four PSI structures and the transport stream. Other examples are possible, but the figure shows the primary connections.

In the following subclauses, each PSI table is described.

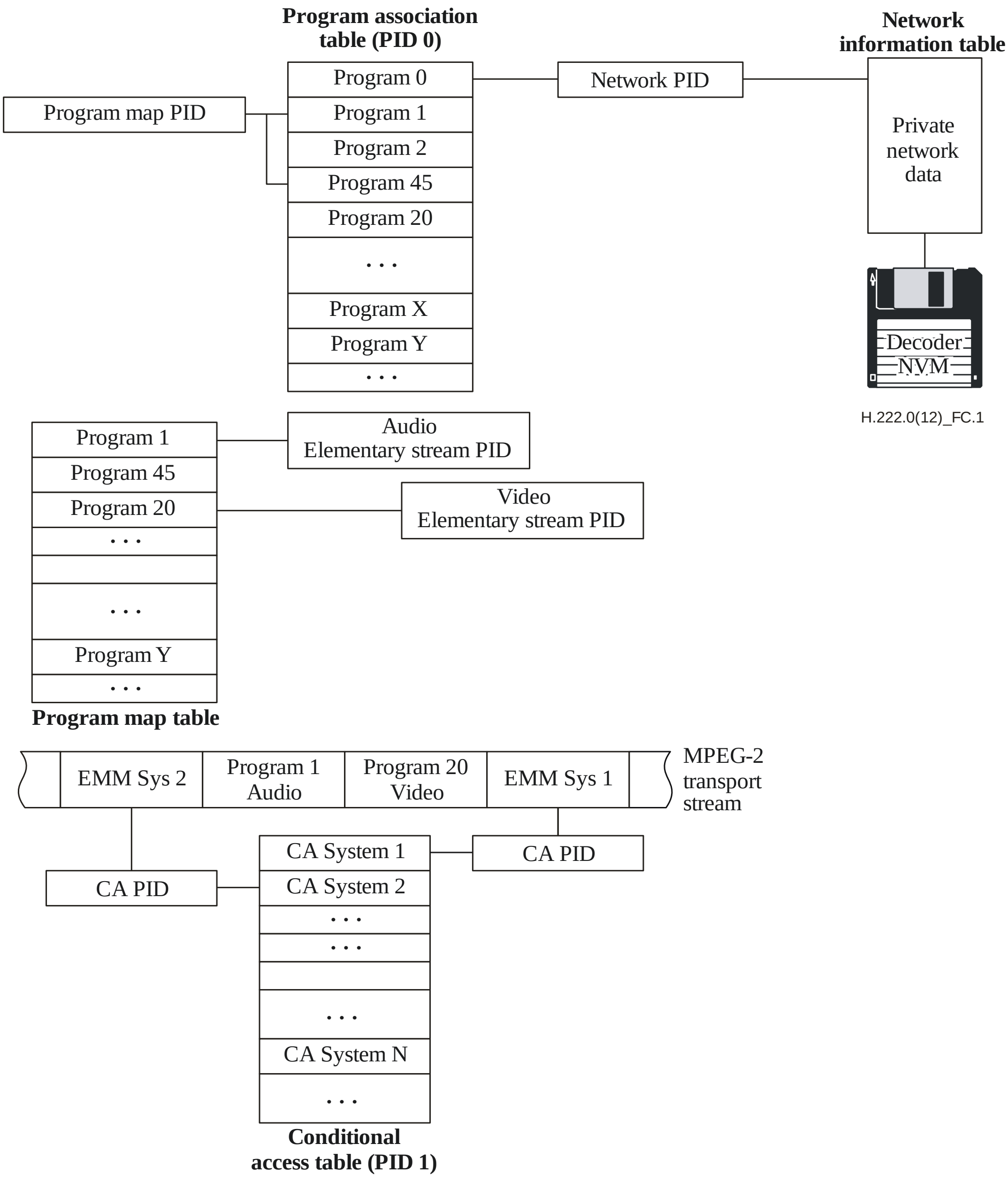


Figure C.1 – Program and network mapping relationships

C.9.1 Program Association Table

Every transport stream must contain a complete valid Program Association Table. The Program Association Table gives the correspondence between a program_number and the PID of the transport stream packets that carry the definition of that program (the PMT_PID). The PAT may be partitioned into up to 255 sections before it is mapped into transport stream packets. Each section carries a part of the overall PAT. This partitioning may be desirable to minimize data loss in error conditions. That is, packet loss or bit errors may be localized to smaller sections of the PAT, thus allowing other sections to still be received and correctly decoded. If all PAT information is put into one section, an error causing a changed bit in the table_id, for example, would cause the loss of the entire PAT. However, this is still permitted as long as the section does not extend beyond the 1024-byte maximum length limit.

Program 0 (zero) is reserved and is used to specify the Network PID. This is a pointer to the transport stream packets which carry the network information table.

The program association table is always transmitted without encryption.

C.9.2 Program map table

The program map table (MPT) provides the mapping between a program number and the program elements that comprise it. This table is present in transport stream packets having one or more privately-selected PID values. These transport stream packets may contain other private structures as defined by the table_id field. It is possible to have TS PMT sections referring to different programs carried in transport stream packets having a common PID value.

This Recommendation | International Standard requires a minimum of program identification: program number, PCR PID, stream types and program elements PIDs. Additional information for either programs or elementary streams may be conveyed by use of the descriptor() construct. Refer to C.9.6.

Private data may also be sent in transport stream packets denoted as carrying TS program map table sections. This is accomplished by the use of the private_section(). In a private_section() the application decides whether version_number and current_next_indicator represent the values of these fields for a single section or whether they are applicable to many sections as parts of a larger private table.

NOTE 1 – Transport stream packets containing the Program Map Table are transmitted unencrypted.

NOTE 2 – It is possible to transmit information on events in private descriptors carried within the TS_program_map_section()s.

C.9.3 Conditional access table

The conditional access table (CAT) gives the association between one or more conditional access (CA) systems, their EMM streams and any special parameters associated with them.

NOTE – The (private) contents of the transport stream packets containing EMM and CA parameters if present will, in general, be encrypted (scrambled).

C.9.4 Network information table

The contents of the network information table (NIT) are private and not specified by this Recommendation | International Standard. In general, it will contain mappings of user-selected services with transport_stream_ids, channel frequencies, satellite transponder numbers, modulation characteristics, etc.

C.9.5 Private_section()

Private_sections() can occur in two basic forms, the short version (where only the fields up to and including section_length are included) or the long version (where all the fields up to and including last_section_number are present, and after the private data bytes the CRC_32 field is present).

Private_section()s can occur in PIDs which are labelled as PMT_PIDs or in transport stream packets with other PID values which contain exclusively private_sections(), including the PID allocated to the NIT. If the transport stream packets of the PID carrying the private_section()s are identified as a PID carrying private_sections (stream_type assignment value 0x05), then only private_sections may occur in transport stream packets of that PID value. The sections may be either of the short or long type.

C.9.6 Descriptors

There are several normative descriptors defined in this Recommendation | International Standard. Many more private descriptors may also be defined. All descriptors have a common format: {tag, length, data}. Any privately defined descriptors must adhere to this format. The data portion of these private descriptors are privately defined.

One descriptor (the CA_descriptor()), is used to indicate the location (PID value of transport packets) of ECM data associated with program elements when it is found in a TS PMT section. When found in a CA section it refers to EMMs.

In order to extend the number of private_descriptors available, the following mechanism could be used: A private descriptor_tag could be privately defined to be constructed as a composite descriptor. This entails privately defining a further sub_descriptor as the first field of the private data bytes of the private descriptor. The described structure is as indicated in Tables C.1 and C.2.

Table C.1 – Composite_descriptor

Syntax	No. of bits	Mnemonic
Composite_descriptor(){		
descriptor_tag(privately defined)	**8**	**uimsbf**
descriptor_length	**8**	**uimsbf**
for (i = 0; i < N; i++){		
sub_descriptor()		
}		
}		

Table C.2 – Sub-descriptor

Syntax	No. of bits	Mnemonic
sub_descriptor() {		
sub_descriptor_tag	**8**	**uimsbf**
sub_descriptor_length	**8**	**uimsbf**
for (i = 0; i < N; i++) {		
private_data_byte	**8**	**uimsbf**
}		
}		

C.10 Bandwidth utilization and signal acquisition time

Any implementation of a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bitstream must make reasonable bandwidth demands for PSI information and, in applications where random access is a consideration, should promote fast signal acquisition. This subclause analyses this issue and gives some broadcast application examples.

The packet-based nature of the transport stream allows for the interspersing of PSI information with fine granularity in the multiplexed data. This provides significant flexibility in the construction and transmission of PSI.

Signal acquisition time in a real decoder is dependent on many factors, including: FDM tuning slew time, demultiplexing time, sequence headers, I-frame occurrence rate and scrambling key retrieval and processing.

This subclause examines both the bitrate and signal acquisition time impacts of the PSI syntax in 2.4.4.4 and 2.4.4.9. It is assumed that the conditional access table does not need to be received dynamically at every program change. This assumption is also made of the private EMM streams. This is because these streams do not contain the quickly-varying ECM components used for program element scrambling (encryption).

Also, in the discussion below, the time to acquire and process ECMs has been neglected.

Tables C.3 and C.4 provide bandwidth usage values for a range of transport stream conditions. One axis of the table is the number of programs contained in a single transport stream. The other axis is the frequency with which the PSI information is transmitted in the transport stream.

Table C.3 – Program association table bandwidth usage (bit/s)
Number of programs per transport stream

		1	5	10	32	128
Frequency of PA table Information (s^{-1})	1	1504	1504	1504	1504	4512
	10	15040	15040	15040	15040	45120
	25	37600	37600	37600	37600	112800
	50	75200	75200	75200	75200	225600
	100	150400	150400	150400	150400	451200
NOTE – Since 46 **program_association_sections** fit into one transport packet, the numbers in the table do not change until the last column.						

Table C.4 – Program map table bandwidth usage (bit/s)
Number of programs per transport stream

		1	5	10	32	128
Frequency of PM table Information (s^{-1})	1	1504	1504	3008	7520	28576
	10	15040	15040	30080	75200	285760
	25	37600	37600	75200	188000	714400
	50	75200	75200	150400	376000	1428800
	100	150400	150400	300800	601600	2857600

This frequency will be a key determinant of the component of signal acquisition time due to PSI structures.

Both bandwidth usage tables assume that only the minimum program mapping information is provided. This means that the PID values and stream types are provided with no additional descriptors. All programs in the example are composed of two elementary streams. Program associations are 2 bytes long, while the minimal program map is 26 bytes long. There is additional overhead associated with version numbers, section lengths, etc. This will be on the order of 1-3% of the total PSI bitrate usage in sections of moderate to maximum length (a few hundred bytes to 1024 bytes) and will thus be ignored here.

The above assumptions allow forty-six (46) program associations to map into one Program Association Table transport stream packet (if no adaptation field is present). Similarly, seven (7) TS_program_map_sections fit into a single transport stream packet. It may be noted that to facilitate easy "drop/add" it is possible to transmit only one (1) TS_program_map_section per PMT_PID. This may cause an undesirable increase in PSI bitrate usage, however.

Using a frequency of 25 Hz for the two PSI Tables, yields a worst-case contribution to the signal acquisition time of approximately 80 ms. This would only occur when the required PAT data was "just missed" and then, once the PAT was acquired and decoded, the required PMT data was also "just missed". This doubling of the worst case acquisition time is one disadvantage of the extra level of indirection introduced by the PAT structure. This effect could be reduced by coordinated transmission of related PAT and PMT packets. Presumably, the advantage that this approach offers for "drop/add" re-multiplexing operations is compensatory.

With the 25-Hz PSI frequency, the following examples may be constructed (all examples leave ample allowance for various datalink, FEC, CA and routing overheads):

6-MHz CATV channel

- five 5.2-Mbit/s programs: 26.5 Mbit/s (includes transport overhead)
- total PSI bandwidth: 5.2 kbit/s
- CA bandwidth: 500 kbit/s

  total Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport bandwidth: 27.1 Mbit/s
- PSI Overhead: 0.28%

OC-3 fibre channel (155 Mbit/s)

- 32 3.9-Mbit/s programs: 127.5 Mbit/s (includes transport overhead)
- total PSI bandwidth: 225.6 kbit/s
- CA bandwidth: 500 kbit/s

  total Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport bandwidth: 128.2 Mbit/s
- PSI Overhead: 0.18%

C-band satellite transponder

- 128 256-kbit/s audio programs: 33.5 Mbit/s (includes transport overhead)
- total PSI bandwidth: 826.4 kbit/s
- CA bandwidth: 500 kbit/s

  total Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport bandwidth: 34.7 Mbit/s
- PSI overhead: 2.4% (actually would be lower if only one PID used per program)

As expected, the percent overhead increases for lower-rate services since many more services are possible per transport stream. However, the overhead is not excessive in all cases. Higher transmission rates (than 25 Hz) for the PSI data may be used to decrease the impact on channel acquisition time with only modest bitrate demand increases.

Annex D

Systems timing model and application implications of this Recommendation | International Standard

(This annex does not form an integral part of this Recommendation | International Standard.)

D.1 Introduction

The Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Systems specification includes a specific timing model for the sampling, encoding, encoder buffering, transmission, reception, decoder buffering, decoding, and presentation of digital audio and video in combination. This model is embodied directly in the specification of the syntax and semantic requirements of compliant Rec. ITU-T H.222.0 | ISO/IEC 13818-1 data streams. Given that a decoding system receives a compliant bit stream that is delivered correctly in accordance with the timing model it is straightforward to implement the decoder such that it produces as output high quality audio and video which are properly synchronized. There is no normative requirement, however, that decoders be implemented in such a way as to provide such high quality presentation output. In applications where the data are not delivered to the decoder with correct timing, it may be possible to produce the desired presentation output; however, such capabilities are not in general guaranteed. This informative annex describes the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Systems timing model in detail, and gives some suggestions for implementing decoder systems to suit some typical applications.

D.1.1 Timing model

Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Systems embodies a timing model in which all digitized pictures and audio samples that enter the encoder are presented exactly once each, after a constant end to end delay, at the output of the decoder. As such, the sample rates, i.e., the video frame rate and the audio sample rate, are precisely the same at the decoder as they are at the encoder. This timing model is diagrammed in Figure D.1:

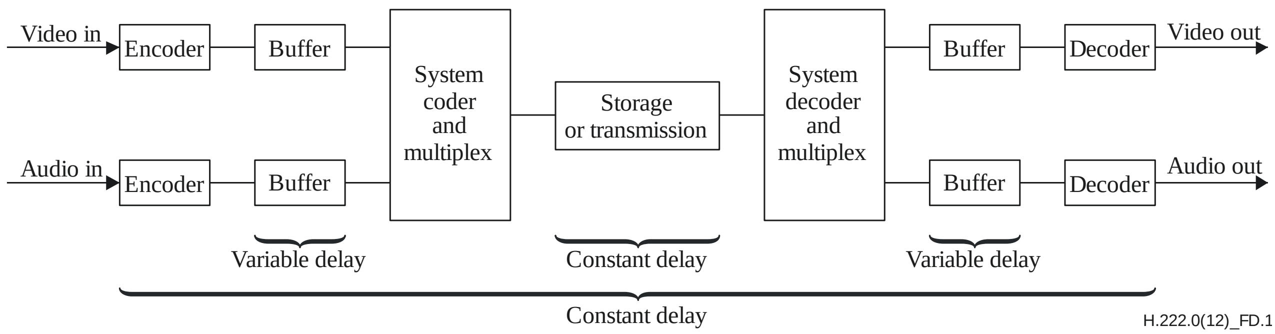


Figure D.1 – Constant delay model

As indicated in Figure D.1, the delay from the input to the encoder to the output or presentation from the decoder is constant in this model[1)], while the delay through each of the encoder and decoder buffers is variable. Not only is the delay through each of these buffers variable within the path of one elementary stream, the individual buffer delays in the video and audio paths differ as well. Therefore the relative location of coded bits representing audio or video in the combined stream does not indicate synchronization information. The relative location of coded audio and video is constrained only by the System Target Decoder (STD) model such that the decoder buffers must behave properly; therefore coded audio and video that represent sound and pictures that are to be presented simultaneously may be separated in time within the coded bit stream by as much as one second, which is the maximum decoder buffer delay that is allowed in the STD model.

The audio and video sample rates at the encoder are significantly different from one another, and may or may not have an exact and fixed relationship to one another, depending on whether the combined stream is a program stream or a transport stream, and on whether the System_audio_locked and System_video_locked flags are set in the program stream. The duration of a block of audio samples (an audio presentation unit) is generally not the same as the duration of a video picture.

---

1) Constant delay as indicated for the entire system is required for correct synchronization, however some deviations are possible. Network delay is discussed as being constant. Slight deviations may be tolerated, and network adaptation may allow greater variations of network delay. Both of these are discussed later.

There is a single, common system clock in the encoder, and this clock is used to create timestamps that indicate the correct presentation and decoding timing of audio and video, as well as to create timestamps that indicate the instantaneous values of the system clock itself at sampled intervals. The timestamps that indicate the presentation time of audio and video are called Presentation Time Stamps (PTS). Those that indicate the decoding time are called Decoding Timestamps (DTS), and those that indicate the value of the system clock are called the System Clock Reference (SCR) in program streams and the Program Clock Reference (PCR) in transport streams. It is the presence of this common system clock in the encoder, the timestamps that are created from it, and the recreation of the clock in the decoder and the correct use of the timestamps that provide the facility to synchronize properly the operation of the decoder.

Encoder implementations may not follow this model exactly; however, the data stream which results from the actual encoder, storage system, network, and one or more multiplexor must follow the model precisely. (Delivery of the data may deviate somewhat, depending on the application). Therefore in this annex, the term "encoder system clock" is used to mean either the actual common system clock as described in this model or the equivalent function, however it may be implemented.

Since the end-to-end delay through the entire system is constant, the audio and video presentations are precisely synchronized. The construction of System bit streams is constrained such that when they are decoded by a decoder that follows this model with the appropriately sized decoder buffers, those buffers are guaranteed never to overflow nor underflow, with specific exceptions allowing intentional underflow.

In order for the decoder system to incur the precise amount of delay that causes the entire end-to-end delay to be constant, it is necessary for the decoder to have a system clock whose frequency of operation and absolute instantaneous value match those of the encoder. The information necessary to convey the encoder's system clock is encoded in the SCR or PCR; this function is explained below.

Decoders which are implemented in accordance with this timing model such that they present audio samples and video pictures exactly once (with specific intentionally coded exceptions), at a constant rate, and such that decoder buffers behave as in the model, are referred to in this annex as precisely timed decoders, or those that produce precisely timed output. Decoder implementations are not required by this International Standard to present audio and video in accordance with this model; it is possible to construct decoders that do not have constant delay, or equivalently do not present each picture or audio sample exactly once. In such implementations, however, the synchronization between presented audio and video may not be precise, and the behaviour of the decoder buffers may not follow the reference decoder model. It is important to avoid overflow at the decoder buffers, as overflow causes a loss of data that may have significant effects on the resulting decoding process. This annex covers primarily the operation of such precisely timed decoders and some of the options that are available in implementing these decoders.

D.1.2 Audio and video presentation synchronization

Within the coding of this Recommendation | International Standard Systems data are timestamps concerning the presentation and decoding of video pictures and blocks of audio samples. The pictures and blocks are called "Presentation Units", abbreviated PU. The sets of coded bits which represent the PUs and which are included within the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 bit stream are called "Access Units", abbreviated AU. An audio access unit is abbreviated AAU, and a video access unit is abbreviated VAU. In ISO/IEC 13818-3 audio the term "audio frame" has the same meaning as AAU or APU (audio presentation unit) depending on the context. A video presentation unit (VPU) is a picture, and a VAU is a coded picture.

Some, but not necessarily all, AAUs and VAUs have associated with them PTSs. A PTS indicates the time that the PU which results from decoding the AU which is associated with the PTS should be presented to the user. The audio PTSs and video PTSs are both samples from a common time clock, which is referred to as the System Time Clock or STC. With the correct values of audio and video PTSs included in the data stream, and with the presentation of the audio and video PUs occurring at the time indicated by the appropriate PTSs in terms of the common STC, precise synchronization of the presented audio and video is achieved at the decoding system. While the STC is not part of the normative content of this Recommendation | International Standard, and the equivalent information is conveyed in this Recommendation | International Standard via such terms as the system_clock_frequency, the STC is an important and convenient element for explaining the timing model, and it is generally practical to implement encoders and decoders which include an STC in some form.

PTSs are required for the conveyance of accurate relative timing between audio and video, since the audio and video PUs generally have significantly different and essentially unrelated duration. For example, audio PUs of 1152 samples each at a sample rate of 44 100 samples per second have a duration of approximately 26.12 ms, and video PUs at a frame rate of 29.97 Hz have a duration of approximately 33.76 ms. In general the temporal boundaries of APUs and VPUs rarely, if ever, coincide. Separate PTSs for audio and video provide the information that indicates the precise temporal relation of audio and video PUs without requiring any specific relationship between the duration and interval of audio and video PUs.

The values of the PTS fields are defined in terms of the System Target Decoder or STD, which is a fundamental normative constraint on all System bit streams. The STD is a mathematical model of an idealized decoder which specifies precisely the movement of all bits into and out of the decoder's buffers, and the basic semantic constraint imposed on the bit stream is that the buffers within the STD must never overflow nor underflow, with specific exceptions provided for underflow in special cases. In the STD model the virtual decoder is always exactly synchronized with the data source, and audio and video decoding and presentation are exactly synchronized. While exact and consistent, the STD is somewhat simplified with respect to physical implementations of decoders in order to clarify its specification and to facilitate its broad application to a variety of decoder implementations. In particular, in the STD model each of the operations performed on the bit stream in the decoder is performed instantaneously, with the obvious exception of the time that bits spend in the decoder buffers. In a real decoder system the individual audio and video decoders do not perform instantaneously, and their delays must be taken into account in the design of the implementation. For example, if video pictures are decoded in exactly one picture presentation interval 1/P, where P is the frame rate, and compressed video data are arriving at the decoder at bit rate R, the completion of removing bits associated with each picture is delayed from the time indicated in the PTS and DTS fields by 1/P, and the video decoder buffer must be larger than that specified in the STD model by R/P. The video presentation is likewise delayed with respect to the STD, and the PTS should be handled accordingly. Since the video is delayed, the audio decoding and presentation should be delayed by a similar amount in order to provide correct synchronization. Delaying decoding and presentation of audio and video in a decoder may be implemented for example by adding a constant to the PTS values when they are used within the decoder.

Another difference between the STD and precise practical decoder implementation is that in the STD model the explicit assumption is made that the final audio and video output is presented to the user instantaneously and without further delay. This may not be the case in practice, particularly with cathode-ray tube displays, and this additional delay should also be taken into account in the design. Encoders are required to encode audio and video such that the correct synchronization is achieved when the data is decoded with the STD. Delays in the input and sampling of audio and video, such as video camera optical charge integration, must be taken into account in the encoder.

In the STD model proper synchronization is assumed and the timestamps and buffer behaviour are tested against this assumption as a condition of bit stream validity. Of course in a physical decoder precise synchronization is not automatically the case, particularly upon start-up and in the presence of timing jitter. Precise decoder timing is a goal to be targeted by decoder designs. Inaccuracy in decoder timing affects the behaviour of the decoder buffers. These topics are covered in more detail in later subclauses of this annex.

The STD includes Decoding Time Stamps (DTS) as well as PTS fields. The DTS refers to the time that an AU is to be extracted from the decoder buffer and decoded in the STD model. Since the audio and video elementary stream decoders are instantaneous in the STD, the decoding time and presentation time are identical in most cases; the only exception occurs with video pictures which have undergone re-ordering within the coded bit stream, i.e., I- and P-pictures in the case of non-low-delay video sequences. In cases where re-ordering exists, a temporary delay buffer in the video decoder is used to store the appropriate decoded I- or P-picture until it should be presented. In all cases where the decoding and presentation times are identical in the STD, i.e., all AAUs, B-picture VAUs, and I- and P-picture VAUs within low-delay video sequences, the DTS is not coded, as it would have the same value as the PTS. Where the values differ, both are coded if either is coded. For all AUs where only the PTS is coded, this field may be interpreted as being both the PTS and the DTS.

Since PTS and DTS values are not required for every AAU and VAU, the decoder may choose to interpolate values which are not coded. PTS values are required with intervals not exceeding 700 ms in each elementary audio and video stream. These time intervals are measured in presentation time, that is, in the same context as the values of the fields, not in terms of the times that the fields are transmitted and received. In cases of data streams where the system, video and audio clocks are locked, as defined in the normative part of this Recommendation | International Standard, each AU following one for which a DTS or PTS is explicitly coded has an effective decoding time of the sum of that for the previous AU plus a fixed and specified difference in value of the STC. For example, in video coded at 29.97 Hz each picture has a difference in time of 3003 cycles of the 90-kHz portion of the STC from the previous picture when the video and system clocks are locked. The same time relationship exists for decoding successive AUs, although re-ordering delay in the decoder affects the relationship between decoder AUs and presented PUs. When the data stream is coded such that the video or audio clock is not locked to the system clock the time difference between decoding successive AUs may be estimated using the same values as indicated above; however, these time differences are not exact due to the fact that relationships between the frame rate, audio sample rate, and system clock frequency were not exact at the encoder.

Note that the PTS and DTS fields do not, by themselves, indicate the correct fullness of the decoder buffers at start up nor at any other time, and equivalently, they do not indicate the amount of time delay that should elapse upon receiving the initial bits of a data stream before decoding should start. This information is retrieved by combining the functions of the PTS and DTS fields and correct clock recovery, which is covered below. In the STD model, and therefore in decoders which are modelled after it, the decoder buffer behaviour is determined completely by the SCR (or PCR) values, the times that they are received, and the PTS and DTS values, assuming that data is delivered in accordance with the timing model. This information specifies the time that coded data spends in the decoder buffers. The amount of data that is in the coded data buffers is not explicitly specified, and this information is not necessary, since the timing is fully specified. Note also that the fullness of the data buffers may vary considerably with time in a fashion that is not predictable by the decoder, except through the proper use of the timestamps.

In order for the audio and video PTSs to refer correctly to a common STC, a correctly timed common clock must be made available within the decoder system. This is subject of the next subclause.

D.1.3 System time clock recovery in the decoder

Within the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Systems data stream there are, in addition to the PTS and DTS fields, clock reference timestamps. These references are samples of the system time clock, which are applicable both to a decoder and to an encoder. They have a resolution of one part in 27 000 000 per second, and occur at intervals up to 100 ms in transport streams, or up to 700 ms in program streams. As such, they can be utilized to implement clock reconstruction control loops in decoders with sufficient accuracy for all identified applications.

In the program stream, the clock reference field is called the System Clock Reference or SCR. In the transport stream, the clock reference field is called the Program Clock Reference or PCR. In general the SCR and PCR definitions may be considered to be equivalent, although there are distinctions. The remainder of this subclause uses the term SCR for clarity; the same statements apply to the PCR except where otherwise noted. The PCR in transport streams provides the clock reference for one program, where a program is a set of elementary streams that have a common time base and are intended for synchronized decoding and presentation. There may be multiple programs in one transport stream, and each may have an independent time base and a separate set of PCRs.

The SCR field indicates the correct value of the STC when the SCR is received at the decoder. Since the SCR occupies more than one byte of data, and System data streams are defined as streams of bytes, the SCR is defined to arrive at the decoder when the last byte of the system_clock_reference_base field is received at the decoder. Alternatively the SCR can be interpreted as the time that the SCR field should arrive at the decoder, assuming that the STC is already known to be correct. Which interpretation is used depends on the structure of the application system. In applications where the data source can be controlled by the decoder, such as a locally attached DSM, it is possible for the decoder to have an autonomous STC frequency, and so the STC need not be recovered. In many important applications, however, this assumption cannot be made correctly. For example, consider the case where a data stream is delivered simultaneously to multiple decoders. If each decoder has its own autonomous STC with its own independent clock frequency, the SCRs cannot be assured to arrive at the correct time at all decoders; one decoder will in general require the SCRs sooner than the source is delivering them, while another requires them later. This difference cannot be made up with a finite size data buffer over an unbounded length of time of data reception. Therefore the following addresses primarily the case where the STC must slave its timing to the received SCRs (or PCRs).

In a correctly constructed and delivered Rec. ITU-T H.222.0 | ISO/IEC 13818-1 data stream, each SCR arrives at the decoder at precisely the time indicated by the value of that SCR. In this context, "time" means correct value of the STC. In concept, this STC value is the same value that the encoder's STC had when the SCR was stored or transmitted. However, the encoding may have been performed not in real time or the data stream may have been modified since it was originally encoded, and in general the encoder or data source may be implemented in a variety of ways such that the encoder's STC may be a theoretical quantity.

If the decoder's clock frequency matches exactly that of the encoder, then the decoding and presentation of video and audio will automatically have the same rate as those at the encoder, and the end-to-end delay will be constant. With matched encoder and decoder clock frequencies, any correct SCR value can be used to set the instantaneous value of the decoder's STC, and from that time on the decoder's STC will match that of the encoder without the need for further adjustment. This condition remains true until there is a discontinuity of timing, such as the end of a program stream or the presence of a discontinuity indicator in a transport stream.

In practice a decoder's free-running system clock frequency will not match the encoder's system clock frequency which is sampled and indicated in the SCR values. The decoder's STC can be made to slave its timing to the encoder using the received SCRs. The prototypical method of slaving the decoder's clock to the received data stream is via a phase-locked loop (PLL). Variations of a basic PLL, or other methods, may be appropriate, depending on the specific application requirements.

A straight forward PLL which recovers the STC in a decoder is diagrammed and described here.

Figure D.2 shows a classic PLL, except that the reference and feedback terms are numbers (STC and SCR or PCR values) instead of signal events such as edges.

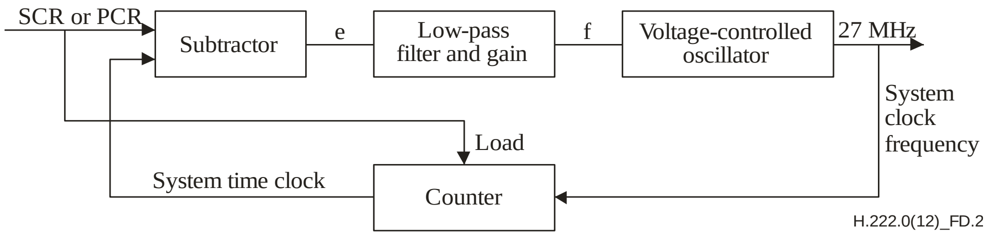


Figure D.2 – STC recovery using PLL

Upon initial acquisition of a new time base, i.e., a new program, the STC is set to the current value encoded in the SCRs. Typically the first SCR is loaded directly into the STC counter, and the PLL is subsequently operated as a closed loop. Variations on this method may be appropriate, i.e., if the values of the SCRs are suspect due to jitter or errors.

The closed-loop action of the PLL is as follows. At the moment that each SCR (or PCR) arrives at the decoder, that value is compared with the current value of the STC. The difference is a number, which has one part in units of 90 kHz and one part in terms of 300 times this frequency, i.e., 27 MHz. The difference value is linearized to be in a single number space, typically units of 27 MHz, and is called "**e**", the error term in the loop. The sequence of **e** terms is input to the low-pass filter and gain stage, which are designed according to the requirements of the application. The output of this stage is a control signal "**f**" which controls the instantaneous frequency of the Voltage Controlled Oscillator (VCO). The output of the VCO is an oscillator signal with a nominal frequency of 27-MHz; this signal is used as the system clock frequency within the decoder. The 27-MHz clock is input to a counter which produces the current STC values, which consist of both a 27-MHz extension, produced by dividing by 300, and a 90-kHz base value which is derived by counting the 90-kHz results in a 33-bit counter. The 33-bit, 90-kHz portion of the STC output is used as needed for comparison with PTS and DTS values. The complete STC is also the feedback input to the subtractor.

The bounded maximum interval between successive SCRs (700 ms) or PCRs (100 ms) allows the design and construction of PLLs which are known to be stable. The bandwidth of the PLLs has an upper bound imposed by this interval. As shown below, in many applications the PLL required has a very low bandwidth, and so this bound typically does not impose a significant limitation on the decoder design and performance.

If the free-running or initial frequency of the VCO is close enough to the correct encoder's system clock frequency, the decoder may be able to operate satisfactorily as soon as the STC is initialized correctly, before the PLL has reached a defined locked state. For a given decoder STC frequency which differs by a bounded amount from the frequency encoded in the SCRs and which is within the absolute frequency bounds required by the decoder application, the effect of the mismatch between the encoder's and the decoder's STC frequencies if there were not PLL is the gradual and unavoidable increase or decrease of the fullness of the decoder's buffers, such that overflow or underflow would occur eventually with any finite size of decoder buffers. Therefore the amount of time allowable before the decoder's STC frequency is locked to that of the encoder is determined by the allowable amount of additional decoder buffer size and delay.

If the SCRs are received by the decoder with values and timing that reflect instantaneously correct samples of a constant frequency STC in the encoder, then the error term **e** converges to an essentially constant value after the loop has reached the locked state. This condition of correct SCR values is synonymous with either constant-delay storage and transmission of the data from the encoder to the decoder, or if this delay is not constant, the effective equivalent of constant delay storage and transmission with the SCR values having been corrected to reflect the variations in delay. With the values of **e** converging to a constant, variations in the instantaneous VCO frequency become essentially zero after the loop is locked; the VCO is said to have very little jitter or frequency slew. While the loop is in the process of locking, the rate of change of the VCO frequency, the frequency slew rate, can be controlled strictly by the design of the low pass filter and gain stage. In general the VCO slew rate can be designed to meet application requirements, subject to constraints of decoder buffer size and delay.

D.1.4 SCR and PCR jitter

If a network or a transport stream re-multiplexor varies the delay in delivering the data stream from the encoder or storage system to the decoder, such variations tend to cause a difference between the values of the SCRs (or PCRs) and the values that they should have when they are actually received. This is referred to as SCR or PCR jitter. For example, if the delay in delivering one SCR is greater than the delay experienced by other similar fields in the same program, that SCR is late. Similarly, if the delay is less than for other clock reference fields in the program, the field is early.

Timing jitter at the input to a decoder is reflected in the combination of the values of the SCRs and the times when they are received. Assuming a clock recovery structure as illustrated in Figure D.2, any such timing jitter will be reflected in the values of the error term **e**; and non-zero values of **e** induce variations in the values of **f**, resulting in variations in the frequency of the 27-MHz system clock. Variations in the frequency of the recovered clock may or may not be acceptable within decoder systems, depending on the specific application requirements. For example, in precisely timed decoders that produce composite video output, the recovered clock frequency is typically used to generate the composite video sample clock and the chroma sub-carrier; the applicable specifications for sub-carrier frequency stability may permit only very slow adjustment of the system clock frequency. In applications where a significant amount of SCR or PCR jitter is present at the decoder input and there are tight constraints on the frequency slew rate of the STC, the constraints of reasonable additional decoder buffer size and delay may not allow proper operation.

The presence of SCR or PCR jitter may be caused for example by network transmission which incorporates packet or cell multiplexing or variable delay of packets through the network, as may be caused by queuing delays or by variable network access time in shared-media systems.

Multiplexing or re-multiplexing of transport or program streams changes the order and relative temporal location of data packets and therefore also of SCRs or PCRs. The change in temporal location of SCRs causes the value of previously correct SCRs to become incorrect, since in general the time at which they are delivered via a constant delay network is not correctly represented by their values. Similarly, a program stream or transport stream with correct SCRs or PCRs may be delivered over a network which imposes a variable delay on the data stream, without correcting the SCR or PCR values. The effect is once again SCR or PCR jitter, with attendant effects on the decoder design and performance. The worst case amount of jitter which is imposed by a network on the SCRs or PCRs received at a decoder depends on a number of factors which are beyond the scope of this Recommendation | International Standard, including the depth of queues implemented in each of the network switches and the total number of network switches or re-multiplexing operations which operate in cascade on the data stream.

In the case of a transport stream, correction of PCRs is necessary in a re-multiplex operation, creating a new transport stream from one or more transport streams. This correction is accomplished by adding a correction term to the PCR; this term can be computed as:

$$\Delta PCR = del_{act} - del_{const}$$

where del_{act} is the actual delay experienced by the PCR, and del_{const} is a constant which is used for all PCRs of that program. The value which should be used for del_{const} will depend on the strategy used by the original encoder/multiplexor. This strategy could be, for instance, to schedule packets as early as possible, in order to allow later transmission links to delay them. In Table D.1, three different multiplex strategies are shown together with the appropriate value for del_{const}.

Table D.1 – Re-multiplexing strategy

Strategy	del_{const}
Early	del_{min}
Late	del_{max}
Middle	del_{avg}

When designing a system, private agreements may be needed as to what strategy should be used by the encoder/multiplexors, since this will have an effect on the ability to perform any additional re-multiplexing.

The amount of multiplex jitter allowed is not normatively bounded in this Recommendation | International Standard. However, 4 ms is intended to be the maximum amount of jitter in a well-behaved system.

In systems which include re-multiplexors special care might be necessary to ensure that the information in the transport stream is consistent. In particular, this applies to PSI and to discontinuity points. Changes in PSI tables might need to be inserted into a transport stream in such a way that subsequent re-multiplexor steps never move them so far that information becomes incorrect. For instance, a new version of PMT section in some cases should not be sent within 4 ms of the data affected by the change.

Similarly, it may be necessary for an encoder/mux to avoid inserting PTS or DTS in a ±4-ms window around a discontinuity point.

D.1.5 Clock recovery in the presence of network jitter

In applications in which there is any significant amount of jitter present in the received clock reference timestamps, there are several choices available for decoder designs; how the decoder is designed depends in large part on the requirements for the decoder's output signal characteristics as well as the characteristics of the input data and jitter.

Decoders in various applications may have differing requirements for the accuracy and stability of the recovered system clock, and the degree of this stability and accuracy that is required may be considered to fall along a single axis. One extreme of this axis may be considered to be those applications where the reconstructed system clock is used directly to synthesize a chroma sub-carrier for use in composite video. This requirement generally exists where the presented video is of the precisely timed type, as described above, such that each coded picture is presented exactly once, and where the output is composite video in compliance with the applicable specifications. In that case the chroma sub-carrier, the pixel clock, and the frame rate all have exactly specified ratios, and all of these have a defined relationship to the system clock. The composite video sub-carrier must have at least sufficient accuracy and stability that any normal television receiver's chroma sub-carrier PLL can lock to the sub-carrier, and the chroma signals which are demodulated using the recovered sub-carrier do not show visible chrominance phase artefacts. The requirement in some applications is to use the system clock to generate a sub-carrier that is in full compliance with the NTSC, PAL, or SECAM specifications, which are typically even more stringent than those imposed by typical television receivers. For example, the SMPTE specification for NTSC requires a sub-carrier accuracy of 3 ppm, with a maximum short term jitter of 1 ns per horizontal line time and a maximum long term drift of 0.1 Hz per second.

In applications where the recovered system clock is not used to generate a chroma sub-carrier, it may still be used to generate a pixel clock for video and it may be used to generate a sample clock for audio. These clocks have their own stability requirements that depend on the assumptions made about the receiving display monitor and on the acceptable amount of audio frequency drift, or "wow and flutter", at the decoder's output.

In applications where each picture and each audio sample are not presented exactly once, i.e., picture and audio sample "slipping" is allowed, the system clock may have relatively loose accuracy and stability requirements. This type of decoder may not have precise audio-video presentation synchronization, and the resulting audio and video presentation may not have the same quality as for precisely timed decoders.

The choice of requirements for the accuracy and stability of the recovered system clock is application dependent. The following focuses on the most stringent requirement which is identified above, i.e., where the system clock is to be used to generate a chroma sub-carrier.

D.1.6 System clock used for chroma sub-carrier generation

The decoder design requirements can be determined from the requirements on the resulting sub-carrier and the maximum amount of network jitter that must be accepted. Similarly, if the system clock performance requirements and the decoder design's capabilities are known, the tolerable maximum network jitter can be determined. While it is beyond the scope of this Recommendation | International Standard to state such requirements, the numbers which are needed to specify the design are identified in order to clarify the statement of the problem and to illustrate a representative design approach.

With a clock recovery PLL circuit as illustrated in Figure D.2, the recovered system clock must meet the requirements of a worst case frequency deviation from the nominal, measured in units of ppm (parts per million), and a worst case frequency slew rate, measured in ppm/s (ppm per second). The peak-to-peak uncorrected network timing jitter has a value that may be specified in milliseconds. In such a PLL the network timing jitter appears as the error term **e** in the diagram, and since the PLL acts as a low-pass filter on jitter at its input, the worst case effect on the 27-MHz output frequency occurs when there is a maximum amplitude step function of PCR timing at the input. The value **e** then has a maximum amplitude equal to the peak-to-peak jitter, which is represented numerically as the jitter times 2**33 in the base portion of the SCR or PCR encoding. The maximum rate of change of the output of the low pass filter (LPF), **f**, with this maximum value of **e** at its input, directly determines the maximum frequency slew rate of the 27-MHz output. For any given maximum value of **e** and maximum rate of change of **f** a LPF can be specified. However, as the gain or cut-off frequency of the LPF is reduced, the time required for the PLL to lock to the frequency represented by the SCRs or PCRs is increased. Implementation of PLLs with very long time constants can be achieved through the use of digital LPF techniques, and possibly analogue filter techniques. With digital LPF implementations, when the frequency term **f** is the input to an analogue VCO, **f** is quantized by a digital to analogue converter, whose step size should be considered when calculating the maximum slew rate of the output frequency.

In order to ensure that **e** converges to a value that approaches zero, the open loop gain of the PLL must be very high, such as might be implemented in an integrator function in the low-pass filter in the PLL.

With a given accuracy requirement, it may be reasonable to construct the PLL such that the initial operating frequency of the PLL meets the accuracy requirement. In this case the initial 27-MHz frequency before the PLL is locked is sufficiently accurate to meet the stated output frequency requirement. If it were not for the fact that the decoder's buffers would eventually overflow or underflow, this initial system clock frequency would be sufficient for long term operation. However, from the time the decoder begins to receive and decode data until the system clock is locked to the time and clock frequency that is represented by the received SCRs or PCRs, data is arriving at the buffers at a different rate than it is being extracted, or equivalently the decoder is extracting access units at times that differ from those of the System Target Decoder (STD) model. The decoder buffers will continue to become more or less full than those of the STD according to the trajectory of recovered system clock frequency with respect to the encoder's clock frequency. Depending on the relative initial VCO frequency and encoder system clock frequency, decoder buffer fullness is either increasing or decreasing. Assuming this relationship is not known, the decoder needs additional data buffering to allow for either case. The decoder should be constructed to delay all decoding operations by an amount of time that is at least equal to the amount of time that is represented by the additional buffering that is allocated for the case of the initial VCO frequency being greater than the encoder's clock frequency, in order to prevent buffer underflow. If the initial VCO frequency is not sufficiently accurate to meet the stated accuracy requirements, then the PLL must reach the locked state before decoding may begin, and there is a different set of considerations regarding the PLL behaviour during this time and the amount of additional buffering and static delay which is appropriate.

A step function in the input timing jitter which produces a step function in the error term **e** of the PLL in Figure D.2 must produce an output frequency term **f** such that when it is multiplied by the VCO gain the maximum rate of change is less than the specified frequency slew rate. The gain of the VCO is stated in terms of the amount of the change in output frequency with respect to a change in control input. An additional constraint on the LPF in the PLL is that the static value of **e** when the loop is locked must be bounded in order to bound the amount of additional buffering and static decoding delay that must be implemented. This term is minimized when the LPF has very high DC gain.

Clock recovery circuits which differ somewhat from that shown in Figure D.2 may be practical. For example, it may be possible to implement a control loop with a Numerically Controlled Oscillator (NCO) instead of a VCO, wherein the NCO uses a fixed frequency oscillator and clock cycles are inserted or deleted from normally periodic events at the output in order to adjust the decoding and presentation timing. There may be some difficulties with this type of approach when used with composite video, as there is a tendency to cause either problematic phase shifts of the sub-carrier or jitter in the horizontal or vertical scan timing. One possible approach is to adjust the period of horizontal scans at the start of vertical blanking, while maintaining the phase of the chroma sub-carrier.

In summary, depending on the values specified for the requirements, it may or may not be practical to construct a decoder which reconstructs the system clock with sufficient accuracy and stability, while maintaining desired decoder buffer sizes and added decoding delay.

D.1.7 Component video and audio reconstruction

If component video is produced at the decoder output, the requirements for timing accuracy and stability are generally less stringent than is the case for composite video. Typically the frequency tolerance is that which the display deflection circuitry can accept, and the stability tolerance is determined by the need to avoid visible image displacement on the display.

The same principles as illustrated above apply; however, the specific requirements are generally easier to meet.

Audio sample rate reconstruction again follows the same principles; however, the stability requirement is determined by the amount of acceptable long and short term sample rate variation. Using a PLL approach as illustrated in the previous subclause, short term deviation can be made to be very small, and longer term frequency variation is manifested as variation in perceived pitch. Again, once specified bounds on this variation are set specific design requirements can be determined.

D.1.8 Frame slipping

In some applications where precise decoder timing is not required, the decoder's system time clock may not adjust its operating frequency to match the frequency represented by received SCRs (or PCRs); it may have a free-running 27 MHz clock instead, while still slaving the decoder's STC to the received data. In this case the STC value must be updated as needed to match the received SCRs. Updating the STC upon receipt of SCRs causes discontinuities in the STC value. The magnitude of these discontinuities depends upon the difference between the decoder's 27-MHz frequency and the encoder's 27-MHz, i.e., that which is represented by the received SCRs, and upon the time interval between successive received SCRs or PCRs. Since the decoder's 27-MHz system clock frequency is not locked to that of the received data, it cannot be used to generate the video or audio sample clocks while maintaining the precise timing assumptions of presenting each video and audio presentation unit exactly once and of maintaining the same picture and audio presentation rate at the decoder and the encoder, with precise audio and video synchronization. There are multiple possibilities for implementing decoding and presentation systems using this structure.

In one type of implementation, the pictures and audio samples are decoded at the time indicated by the decoder's STC, while they are presented at slightly different times, according to the locally produced sample clocks. Depending on the relationships of the decoder's sample clocks to the encoder's system clock, pictures and audio samples may on occasion be presented more than one each or not at all; this is referred to as "frame slipping" or "sample slipping", in the case of audio. There may be perceptible artefacts introduced by this mechanism. The audio-video synchronization will in general not be precise, due to the units of time over which pictures, and perhaps audio presentation units, are repeated or deleted. Depending on the specific implementation, additional buffering in the decoder is generally needed for coded data or decoded presentation data. Decoding may be performed immediately before presentation, and not quite at the time indicated in the decoder's STC, or decoded presentation units may be stored for delayed and possibly repeated presentation. If decoding is performed at the time of presentation, a mechanism is required to support deleting the presentation of pictures and audio samples without causing problems in the decoding of predictively coded data.

D.1.9 Smoothing of network jitter

In some applications it may be possible to introduce a mechanism between a network and a decoder in order to reduce the degree of jitter which is introduced by a network. Whether such an approach is feasible depends on the type of streams received and the amount and type of jitter which is expected.

Both the transport stream and the program stream indicate within their syntax the rate at which the stream is intended to be input to a decoder. These indicated rates are not precise, and cannot be used to reconstruct data stream timing exactly. They may, however, be useful as part of a smoothing mechanism.

For example, a transport stream may be received from a network such that the data is delivered in bursts. It is possible to buffer the received data and to transmit data from the buffer to the decoder at an approximately constant rate such that the buffer remains approximately one-half full.

However, a variable rate stream should not be delivered at constant rate, and with variable rate streams the smoothing buffer should not always be one-half full. A constant average delay through the buffer requires a buffer fullness that varies with the data rate. The rate that data should be extracted from the buffer and input to the decoder can be approximated using the rate information present in the data stream. In transport streams the intended rate is determined by the values of the PCR fields and the number of transport stream bytes between them. In program streams the intended rate is explicitly specified as the Program_mux_rate, although as specified in this Recommendation | International Standard the rate may drop to zero at SCR locations, i.e., if the SCR arrives before the time expected when the data is delivered at the indicated rate.

In the case of variable rate streams, the correct fullness of the smoothing buffer varies with time, and may not be determined exactly from the rate information. In an alternative approach, the SCRs or PCRs may be used to measure the time when data enter the buffer and to control the time when data leave the buffer. A control loop can be designed to provide constant average delay through the buffer. It may be observed that such a design is similar to the control loop illustrated in Figure D.2. The performance obtainable from inserting such a smoothing mechanism before a decoder can also be achieved by cascading multiple clock recovery PLLs. The rejection of jitter from the received timing will benefit from the combined low pass filter effect of the cascaded PLLs.

Annex E

Data transmission applications

(This annex does not form an integral part of this Recommendation | International Standard.)

E.1 General considerations

- Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport multiplex will be used to transmit data as well as video and audio.
- Data elementary streams are not continuous as may appear video and audio streams in broadcast applications.
- While it is already possible to identify the beginning of a PES packet, it is not always possible to identify the end of a PES packet by the beginning of the next PES packet, because it is possible for one or more Transport packet carrying PES packets to be lost.

E.2 Suggestion

A suitable solution is to transmit the following PES packet just after an associated PES packet. A PES packet without payload may be sent when there are no further PES packets to send.

Table E.1 is an example of such a PES packet.

Table E.1 – PES packet header example

PES packet header fields	Values
packet_start_code_prefix	0x000001
stream_id	assigned
PES_packet_length	0x0003
'10'	'10'
PES_scrambling_control	'00'
PES_priority	'0'
data_alignment_indicator	'0'
copyright	'0'
original_or_copy	'0'
PTS_DTS_flags	'00'
ESCR_flag	'0'
ES_rate_flag	'0'
DSM_trick_mode_flag	'0'
additional_copy_info_flag	'0'
PES_CRC_flag	'0'
PES_extension_flag	'0'
PES_header_data_length	0x00

Annex F

Graphics of syntax for this Recommendation | International Standard

(This annex does not form an integral part of this Recommendation | International Standard.)

F.1 Introduction

This annex is an informative annex presenting graphically the transport stream and program stream syntax. This annex in no way replaces any normative clause(s).

In order to produce clear drawings, not all fields have been fully described or represented. Reserved fields may be omitted or indicated by areas with no detail. Fields lengths are indicated in bits.

F.1.1 Transport stream syntax

See Figure F.1.

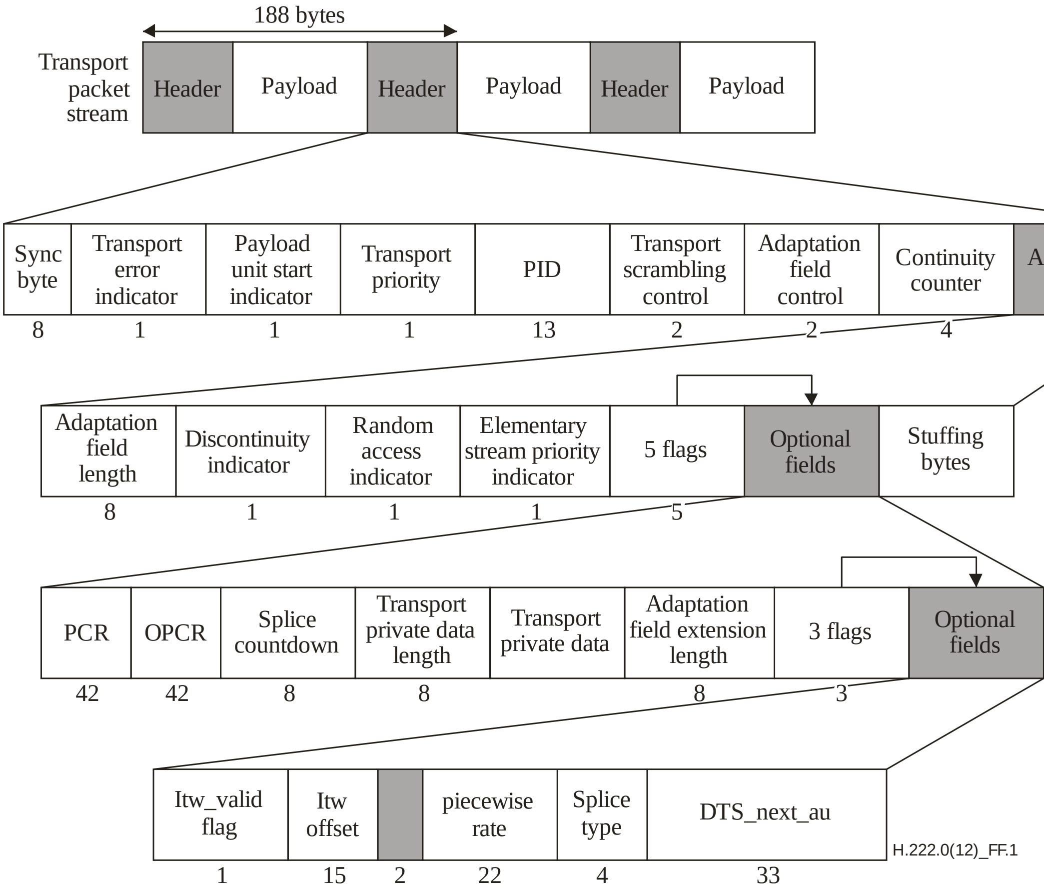


Figure F.1 – Transport stream syntax diagram

F.1.2 PES packet

See Figure F.2.

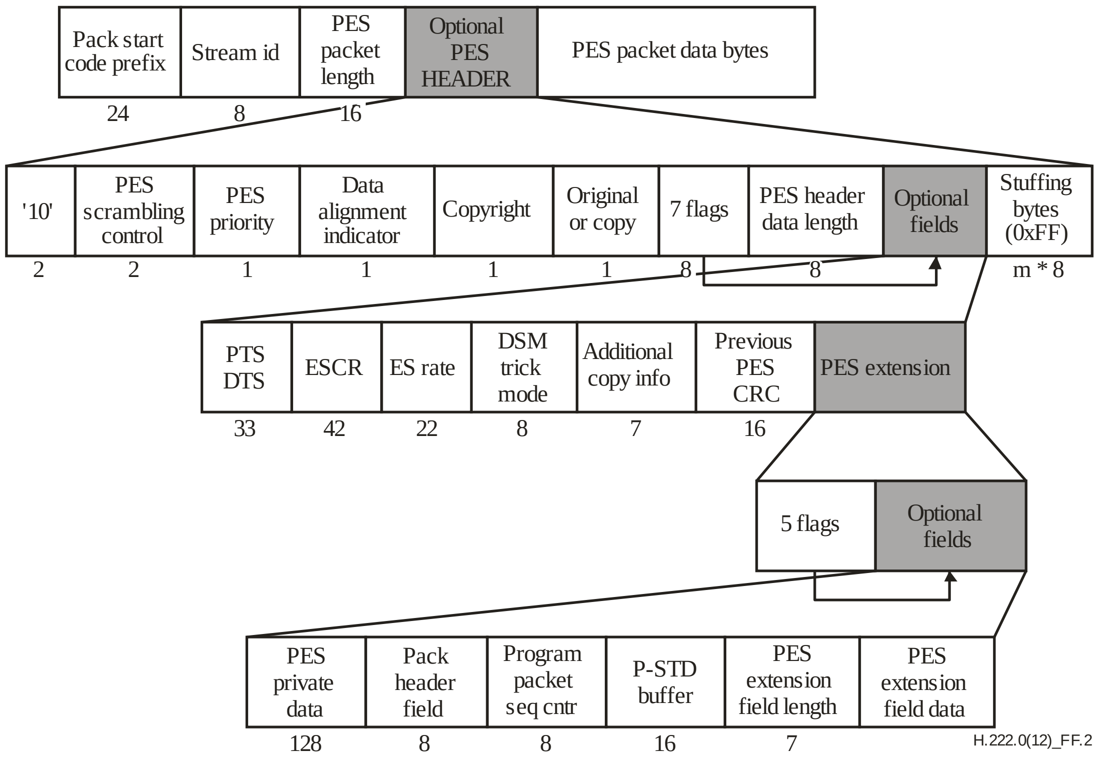


Figure F.2 – PES packet syntax diagram

F.1.3 Program Association section

See Figure F.3.

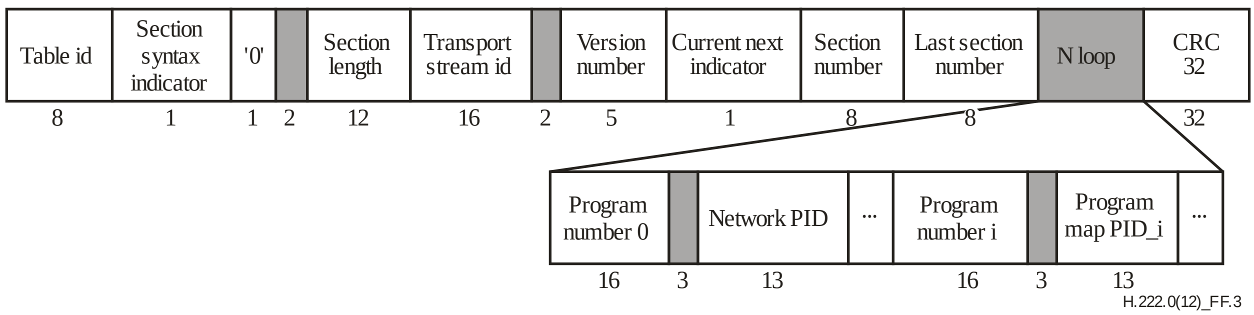


Figure F.3 – Program association section diagram

F.1.4 CA section

See Figure F.4.

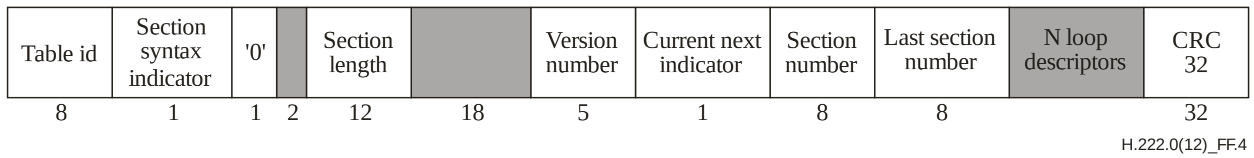


Figure F.4 – Conditional access section diagram

F.1.5 TS program map section

See Figure F.5.

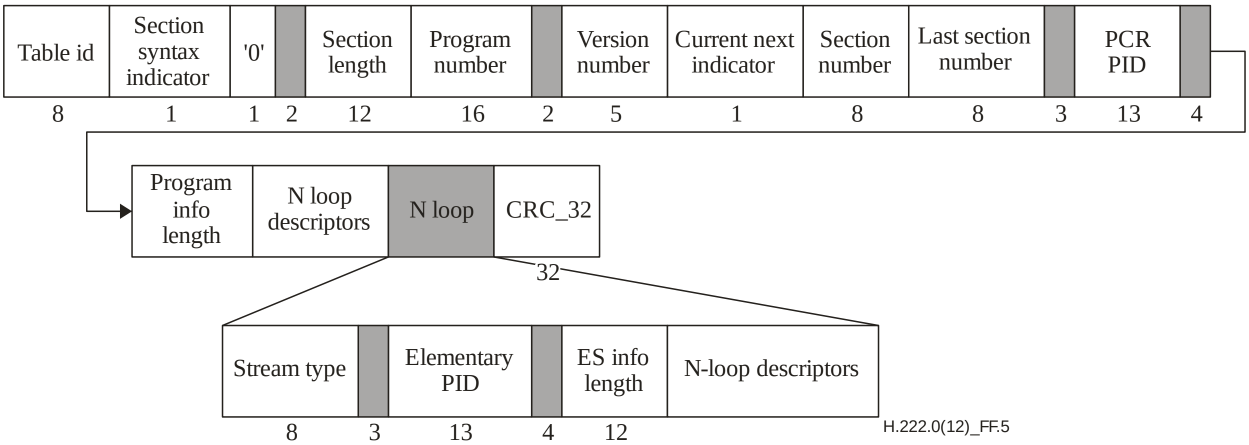


Figure F.5 – TS program map section diagram

F.1.6 Private section

See Figure F.6.

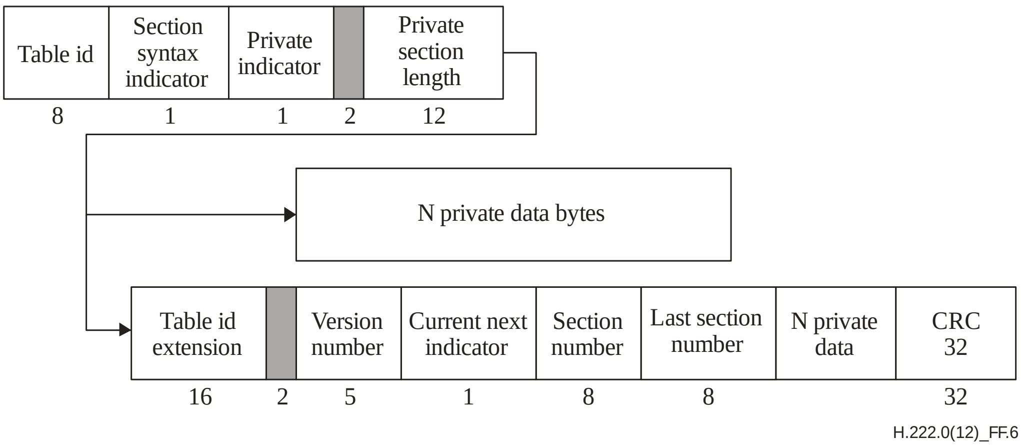


Figure F.6 – Private section diagram

F.1.7 Program stream

See Figure F.7.

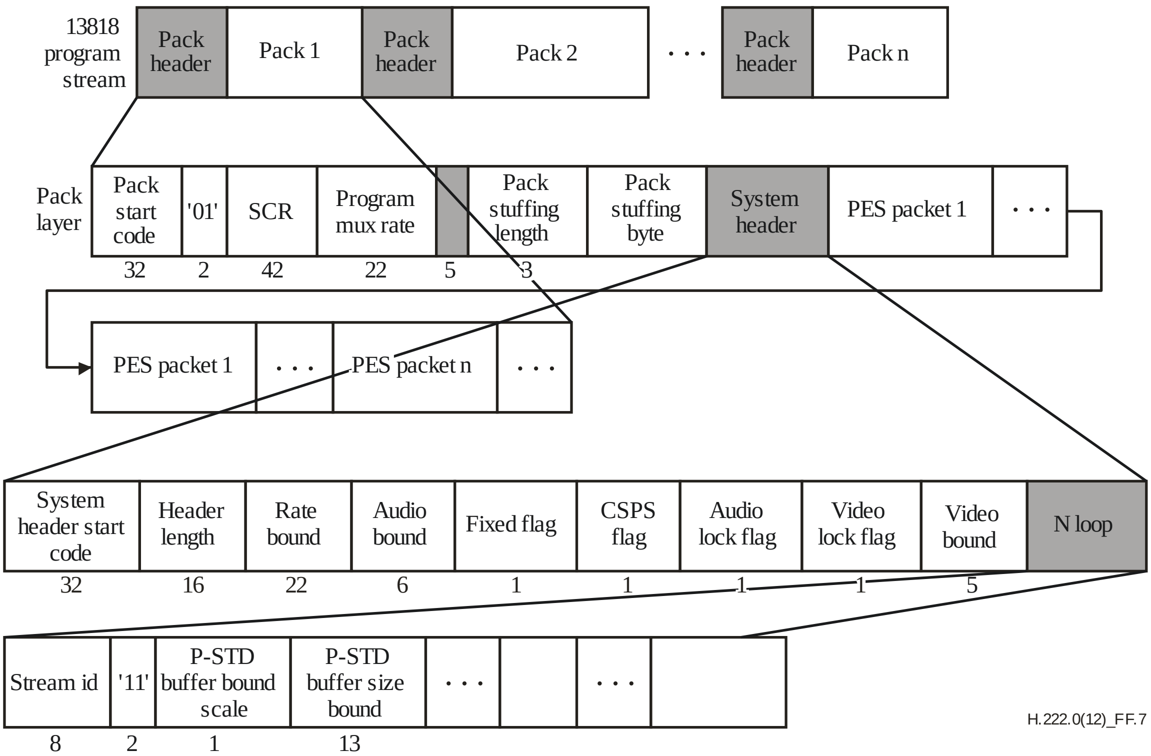


Figure F.7 – Program stream diagram

F.1.8 Program stream map

See Figure F.8.

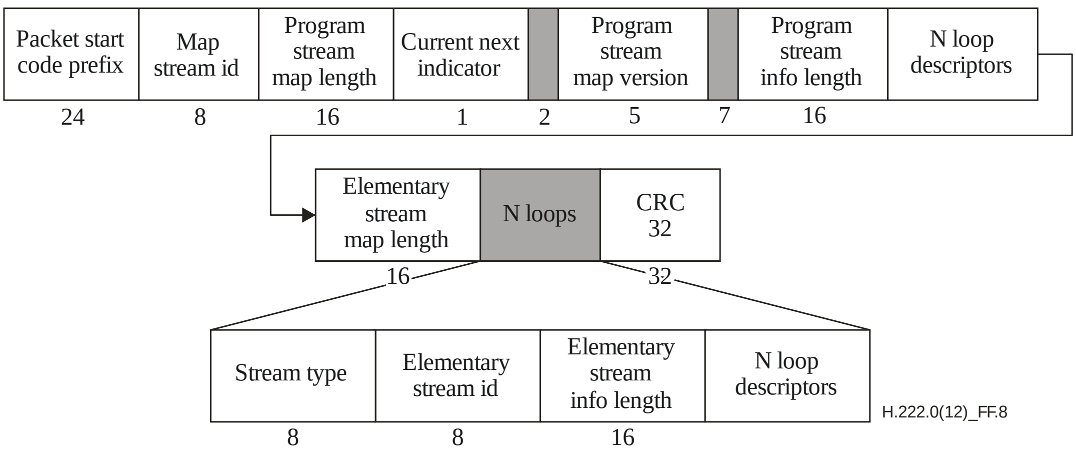


Figure F.8 – Program stream map diagram

Annex G

General information

(This annex does not form an integral part of this Recommendation | International Standard.)

G.1 General information

G.1.1 Sync byte emulation

In the choice of PID values it is recommended that the periodic emulation of sync bytes be avoided. Such emulation may potentially occur within the PID field or as a combination of the PID field and adjacent flag settings. It is recommended that emulation of the sync byte be permitted to occur in the same position of the packet header for a maximum of 4-consecutive transport packets.

G.1.2 Skipped picture status and decoding process

Assume that the sequence being displayed contains only I- and P-frames. Denote the next picture to be decoded by picture_next, and the picture currently being displayed by picture_current. Because of the fact that the video encoder may skip pictures, it is possible that not all of the bits of picture_next are present in the STD buffers EB_n or B_n when the time arrives to remove those bits for instantaneous decoding and display. When this case arises, no bits are removed from the buffer and picture_current is displayed again. When the next picture display time arrives, if the remainder of the bits corresponding to picture_next are now in buffer EB_n or B_n, all the bits of picture_next are removed and picture_next is displayed. If all the bits of picture_next are not in the buffer EB_n or B_n, the above process of re-displaying picture_current is repeated. This process is repeated until picture_next can be displayed. Note that if a PTS preceded picture_next in the bitstream, it will be incorrect by some multiple of the picture display interval, which itself may depend on some parameters, and must be ignored.

Whenever the skipped picture situation described above occurs, the encoder is required to insert a PTS before the picture to be decoded after picture_next. This allows the decoder to immediately verify that it has correctly displayed the received picture sequence.

G.1.3 Selection of PID values

Applications are encouraged to use low numbered PID values (avoiding reserved values as specified in Table 2-4) and group values together as much as possible.

G.1.4 PES start_code emulation

Three consecutive bytes having the value of a packet_start_code_prefix (0x000001), which when concatenated with a fourth byte, may emulate the four bytes of a PES_packet_header at a unintended place in the stream.

Such, so called, start code emulation is not possible in video elementary streams. It is possible in audio and data elementary streams. It is also possible at the boundary of a PES_packet_header and a PES_packet payload, even if the PES_packet payload is video.

Annex H

Private data

(This annex does not form an integral part of this Recommendation | International Standard.)

H.1 Private data

Private data is any user data which is not coded according to a standard specified by ITU-T | ISO/IEC and referred to in this Specification. The contents of this data is not and shall not be specified within this Recommendation | International Standard in the future. The STD defined in this Specification does not cover private data other than the demultiplex process. A private party may define each STD for private streams.

Private data may be carried in the following locations within the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 syntax.

1) *Transport stream packet Table 2-2*

   The data bytes of the transport_packet() syntax may contain private data. Private data carried in this format is referred to as user private within the stream_type Table 2-34. It is permitted for transport stream packets containing private data to also include adaptation_field()s.

2) *Transport stream adaptation field Table 2-6*

   The presence of any optional private_data_bytes in the adaptation_field() is signalled by the transport_private_data_flag. The number of the private_data_bytes is inherently restricted by the semantic of the adaptation_field_length field, where the value of the adaptation_field_length shall not exceed 183 bytes.

3) *PES packet Table 2-21*

   There are two possibilities for carrying private data within PES packets. The first possibility is within the PES_packet_header, within the optional 16 bytes of PES_private_data. The presence of this field is signalled by the PES_private_data_flag. The presence of the PES_private_data_flag is signalled by the PES_extension_flag. If present, these bytes, when considered with the adjacent fields, shall not emulate the packet_start_code_prefix.

   The second possibility is within the PES_packet_data_byte field. This may be referred to as private data within PES packets under the stream_type Table 2-34. This category of private data can be split in two: private_stream_1 refers to private data within PES packets which follow the PES_packet() syntax such that all fields up to and including, but not limited to, PES_header_data_length are present. private_stream_2 refers to private data within PES packets where only the first three fields shall be present followed by the PES_packet_data_bytes containing private data.

   Note that PES packets exist within both program streams and transport streams therefore private_stream_1 and private_stream_2 exist within both program streams and transport streams.

4) *Descriptors*

   Descriptors exist within program streams and transport streams. A range of private descriptors may be defined by the user. These descriptors shall commence with descriptor_tag and descriptor_length fields. For private descriptors, the value of descriptor_tag may take the values 64-255 as identified in Table 2-45. These descriptors may be placed within a program_stream_map() Table 2-34, a CA_section() Table 2-32, a TS_program_map_section(), Table 2-33 and in any private_section(), Table 2-35.

   Specifically private_data_bytes also appear in the CA_descriptor().

5) *Private Section*

   The private_section Table 2-35 provides a further means to carry private data also in two forms. This type of elementary stream may be identified under stream_type Table 2-34 as private_data in PSI sections. One type of private_section() includes only the first five defined fields, and is followed by private data. For this structure the section_syntax_indicator shall be set to a value of '0'. For the other type, the section_syntax_indicator shall be set to a value of '1' and the full syntax up to and including last_section_number shall be present, followed by private_data_bytes and ending with the CRC_32.

Annex I

Systems conformance and real-time interface

(This annex does not form an integral part of this Recommendation | International Standard.)

I.1 Systems conformance and real-time interface

Conformance for Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program streams and transport streams is specified in terms of the normative specifications in this Recommendation | International Standard. These specifications include, among other requirements, a System Target Decoder (T-STD and P-STD) which specifies the behaviour of an idealized decoder when the stream is the input to such a decoder. This model, and the associated verification, do not include information concerning the real-time delivery performance of the stream, except for the accuracy of the system clock frequency which is represented by the transport stream and the program stream. All transport streams and program streams must comply with this Recommendation | International Standard.

In addition, there is a real-time interface specification for input of transport streams and program streams to a decoder. This Recommendation | International Standard allows standardization of the interface between MPEG decoders and adapters to networks, channels, or storage media. The timing effects of channels, and the inability of practical adapters to eliminate completely these effects, causes deviations from the idealized byte delivery schedule to occur. While it is not necessary for all MPEG decoders to implement this interface, implementations which include the interface shall adhere to the specifications. This Recommendation | International Standard covers the real-time delivery behaviour of transport streams and Program streams to decoders, such that the coded data buffers in decoders are guaranteed not to overflow nor underflow, and decoders are guaranteed to be able to perform clock recovery with the performance required by their applications.

The MPEG real-time interface specifies the maximum allowable amount of deviation from the idealized byte delivery schedule which is indicated by the Program Clock Reference (PCR) and System Clock Reference (SCR) fields encoded in the stream.

Annex J

Interfacing jitter-inducing networks to MPEG-2 decoders

(This annex does not form an integral part of this Recommendation | International Standard.)

J.1 Introduction

In this annex the expression "system stream" will be used to refer to both Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport streams and Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program streams. When the term "STD" is used, it is understood to mean the P-STD (Program System Target Decoder) for program streams and the T-STD (Transport System Target Decoder) for transport streams.

The intended byte delivery schedule of a system stream can be deduced by analysing the stream. A system stream is compliant if it can be decoded by the STD, which is a mathematical model of an idealized decoder. If a compliant system stream is transmitted over a jitter-inducing network, the true byte delivery schedule may differ significantly from the intended byte delivery schedule. In such cases it may not be possible to decode the system stream on such an idealized decoder, because jitter may cause buffer overflows or underflows and may make it difficult to recover the time base. An important example of such a jitter-inducing network is ATM.

The purpose of this annex is to provide guidance and insight to entities concerned with sending system streams over jitter-inducing networks. Network-specific compliance models for transporting system streams are likely to be developed for several types of networks, including ATM. The STD plus a real-time interface definition can play an integral role in defining such models. A framework for developing network compliance models is presented in J.2.

Three examples of network encoding to enable the building of jitter-smoothing network adapters are discussed in J.3. In the first example, a constant bitrate system stream is assumed and a FIFO is used for jitter smoothing. In the second example, the network adaptation layer includes timestamps to facilitate jitter smoothing. In the final example, a common network clock is assumed to be available end-to-end, and is exploited to achieve jitter smoothing.

Clause J.4 presents two examples of decoder implementations in which network-induced jitter can be accommodated. In the first example, a jitter-smoothing network adapter is inserted between a network's output and an MPEG-2 decoder. The MPEG-2 decoder is assumed to conform to a real-time MPEG-2 interface specification. This interface requires an MPEG-2 decoder with more jitter tolerance than the idealized decoder of the STD. The network adapter processes the incoming jittered bitstream and outputs a system stream whose true byte delivery schedule conforms to the real-time specification. Example one is discussed in J.4.1. For some applications the network adapter approach will be too costly because it requires two stages of processing. Therefore, in the second example the dejittering and MPEG-2 decoding functions are integrated. The intermediate processing of the jitter-removal device is bypassed, so only a single stage of clock recovery is required. Decoders that perform integrated dejittering and decoding are referred to in this annex as integrated network-specific decoders, or simply integrated decoders. Integrated decoders are discussed in J.4.2.

In order to build either network adapters or integrated decoders a maximum value for the peak-to-peak network jitter must be assumed. In order to promote interoperability, a peak-to-peak jitter bound must be specified for each relevant network type.

J.2 Network compliance models

One way to model the transmission of a system stream across a jitter-inducing network is shown in Figure J.1.

The system stream is input to a network-specific encoding device that converts the system stream into a network-specific format. Information to assist in jitter removal at the network output may be part of this format. The network decoder comprises a network-specific decoder and a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 decoder. The Rec. ITU-T H.222.0 | ISO/IEC 13818-1 decoder is assumed to conform to a real-time interface specification, and could have the same architecture as the STD with appropriate buffers made larger to provide more jitter tolerance. The network-specific decoder removes the non- Rec. ITU-T H.222.0 | ISO/IEC 13818-1 data added by the network-specific encoder and dejitters the network's output. The output of the network-specific decoder is a system stream that conforms to the real-time specification.

A network target decoder (NTD) can be defined based on the above architecture. A compliant network bitstream would be one that was able to be decoded by the NTD. A network decoder would be compliant provided it could decode any network bitstream able to be decoded by the NTD. A real network decoder might or might not have the architecture of the NTD.

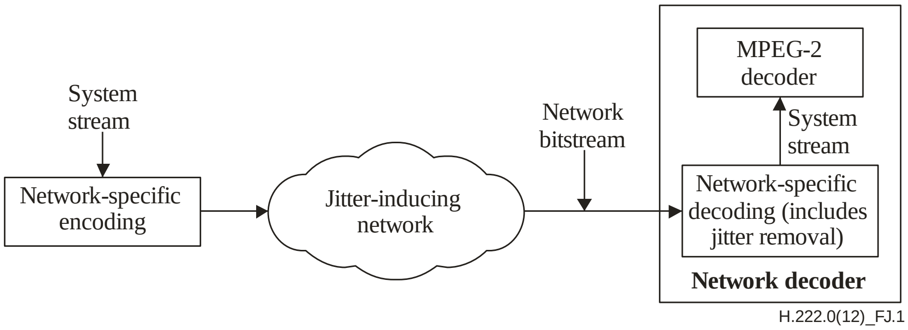


Figure J.1 – Sending system streams over a jitter-inducing network

J.3 Network specification for jitter smoothing

In the case of constant bit rate system streams, jitter smoothing can be accomplished with a FIFO. Additional data that provides specific support for dejittering is not required in the network adaptation layer. After the bytes added by the network encoding are removed, the system stream data is placed in a FIFO. A PLL keeps the buffer approximately half full by adjusting the output rate in response to changes in buffer fullness. In this example the amount of jitter-smoothing achieved will depend on the size of the FIFO and the characteristics of the PLL.

Figure J.2 illustrates a second way to accomplish jitter smoothing. In this example timestamp support from a network adaptation layer is assumed. Using this technique, both constant bit rate and variable bit rate system streams can be dejittered.

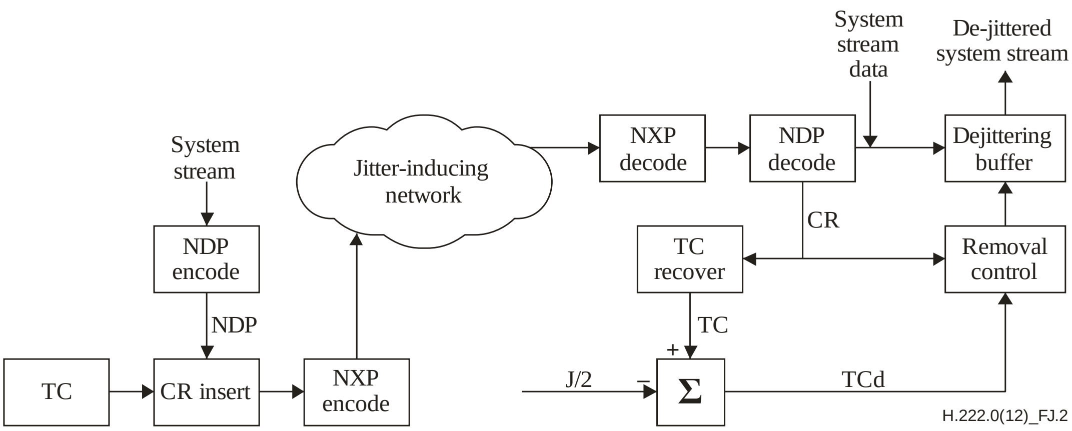


Figure J.2 – Jitter-smoothing using network-layer timestamps

Assume the network adapter is designed to compensate for a peak-to-peak jitter of J seconds. The intended byte delivery schedule is reconstructed using Clock Reference (CRs) samples taken from a Time Clock (TC). The CRs and the TC are analogous to PCRs and the STC. The Network Data Packet (NDP) encode converts each system stream packet into a Network Data Packet (NDP). The network data packets contain a field for carrying CR values, and the current value of the TC is inserted into this field as the NDP leaves the NDP encoder. The Network Transport Packetization (NXP) function encapsulates the NDPs into network transport packets. After transmission across the network, the CRs are extracted by the NDP decoder as the NDPs enter the NDP decoder. The CRs are used to reconstruct the TC, for example by using a PLL. The first MPEG-2 packet is removed from the dejittering buffer when the delayed TC (TCd) is equal to the first MPEG-2 packet's CR. Subsequent MPEG-2 packets are removed when their CR values equal the value of the TCd.

Ignoring implementation details such as the speed of the TC clock recovery loop and the spectral purity of the TC, the size of the dejittering buffer depends only on the maximum peak-to-peak jitter to be smoothed and the largest transport rate that occurs in the system stream. The dejittering buffer size, B_{dj}, is given by

$$B_{dj} = JR_{max}$$

where R_{max} is the maximum data rate of the system stream in bits per second. When packets traversing the network experience the nominal delay, the buffer is half full. When they experience a delay of J/2 seconds, the buffer is empty, and when they experience a delay (advance) of –J/2 seconds the buffer is full.

As a final example, in some cases a common network clock will be available end-to-end, and it may be feasible to lock the system clock frequency to the common clock. The network adapter can smooth jitter with a FIFO. The adapter uses PCRs or SCRs to reconstruct the original byte delivery schedule.

J.4 Example decoder implementations

J.4.1 Network adapter followed by an MPEG-2 decoder

In this implementation a network adapter conforming to the network compliance specification is connected to an MPEG-2 decoder conforming to the real-time interface specification.

J.4.2 Integrated decoder

The example presented in J.4.1 requires two stages of processing. The first stage is necessary to dejitter the network's output. The second stage, recovering the STC by processing PCRs or SCRs, is required for STD decoding. The example presented in this subclause is a decoder that integrates the dejittering and decoding functions in a single system. The STC clock is recovered directly using the jittered PCR or SCR values. For presenting this example, an MPEG-2 transport stream will be assumed.

Figure J.3 illustrates the operation of the integrated decoder. The stream of network packets input to the decoder is assumed to be the same as the one shown in Figure J.2.

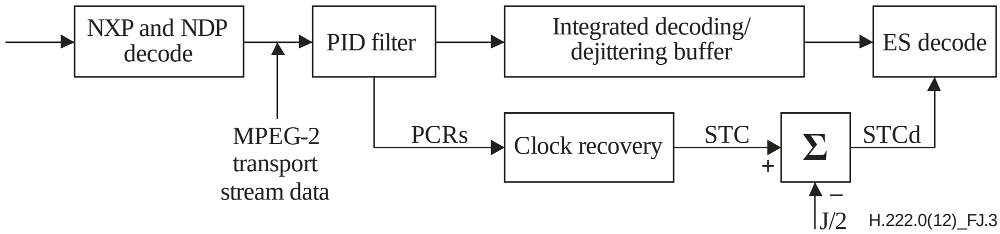


Figure J.3 – Integrated dejittering and MPEG-2 decoding

The incoming network packets are reassembled into MPEG-2 transport stream data by the NXP and NDP decode functions. The jittered Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport stream packets are then filtered to extract packets with the desired PID. For the case illustrated, the PID being decoded is also carrying the PCRs. The PCR values are sent to a PLL to recover the STC. Entire packets for the selected PID are placed in the integrated buffer. A positive value of J/2 s is subtracted from the STC to obtain the delayed STC, STCd. Again, J is the peak-to-peak jitter the network-savvy decoder can accommodate. The delay is introduced to guarantee that all the data required for an access unit has arrived in the buffer when the PTS/DTS of the access unit equals the current value of the STCd.

Ignoring implementation details such as the speed of the STC clock recovery loop and the spectral purity of the STC:

$$\begin{aligned} B_{size} &= B_{dec} + B_{mux} + B_{OH} + 512 + B_j \\ &= B_n + 512 + B_j \end{aligned}$$

where $B_j = R_{max}\ J$ and R_{max} is the maximum rate at which data is input to the PID filter. Depending on the implementation, the integrated memory could be broken into two components as in the transport STD.

Annex K

Splicing transport streams

(This annex does not form an integral part of this Recommendation | International Standard.)

K.1 Introduction

For the purposes of this annex, the term 'splicing' refers to the concatenation performed on the Transport level of two different elementary streams, the resulting transport stream conforming totally to this Recommendation | International Standard. The two elementary streams may have been generated at different locations and/or at different times, and were not necessarily intended to be spliced together when they were generated. In the following we will call the 'old' stream a continuous elementary stream (video or audio), which has been superseded by another stream (the 'new' one) from a certain point on. This point is called the splice. It is the boundary between data belonging to the 'old' stream and data belonging to the 'new' one.

A splice can be seamless or non-seamless:

- A seamless splice is a splice inducing no decoding discontinuity (refer to 2.7.6). This means that the decoding time of the first access unit of the 'new' stream is consistent with respect to the decoding time of the access unit of the 'old' stream preceding the splice, i.e., it is equal to the one that the next access unit would have had if the 'old' stream had continued. In the following, we will call this decoding time the 'seamless decoding time'.
- A non-seamless splice is a splice which results in a decoding discontinuity, i.e., the decoding time of the first access unit of the 'new' stream is greater than the seamless decoding time.

  NOTE – A decoding time lower than the seamless decoding time is forbidden.

Splicing is allowed to be performed at any transport stream packet boundary, since the resulting stream is legal. But in a general case, if nothing is known about the location of PES packet starts and access unit starts, this constraint imposes that not only the Transport layer is parsed, but also the PES layer and the Elementary Stream layer, and may in some cases, make some processing on the payload of transport stream packets necessary. If such complex operations are wished to be avoided, splicing should be performed at locations where the transport stream has favourable properties, these properties being indicated by the presence of a splicing point.

The presence of a splicing point is indicated by the splice_flag and splice_countdown fields (refer to 2.4.3.4 for the semantics of these fields). In the following, the transport stream packet in which the splice_countdown field value reaches zero will be called 'splicing packet'. The splicing point is located immediately after the last byte of the splicing packet.

K.2 The different types of splicing point

A splicing point can be either an ordinary splicing point or a seamless splicing point.

K.2.1 Ordinary splicing points

If the seamless_splice_flag field is not present, or if its value is zero, the splicing point is ordinary. The presence of an ordinary splicing point only signals alignment properties of the Elementary Stream: the splicing packet ends on the last byte of an Access Unit, and the payload of the next transport stream packet of the same PID will start with the header of a PES packet, the payload of which will start with an Elementary Stream Access Point (or with a sequence_end_code() immediately followed by an Elementary Stream Access Point, in the case of video). These properties allow 'Cut and Paste' operations to be performed easily on the Transport level, while respecting syntactical constraints and ensuring bit stream consistency. However, it does not provide any information concerning timing or buffer properties. As a consequence, with such splicing points, seamless splicing can only be done with the help of private arrangements, or by analysing the payload of the transport stream packets and tracking buffer status and timestamp values.

K.2.2 Seamless splicing points

If the seamless_splice_flag field is present and its value is one, information is given by the splicing point, indicating some properties of the 'old' stream. This information is not aimed at decoders. Its primary goal is to facilitate seamless splicing. Such a splicing point is called a seamless splicing point. The available information is:

- The seamless decoding time, which is encoded as a DTS value in the DTS_next_AU field. This DTS value is expressed in the time base which is valid in the splicing packet.

- In the case of a video elementary stream, the constraints that have been applied to the 'old' stream when it was generated, aiming at facilitating seamless splicing. These conditions are given by the value of the splice_type field, in the table corresponding to the profile and level of the video stream.

Note that a seamless splicing point can be used as an ordinary splicing point, by discarding this additional information. This information may also be used if judged helpful to perform non-seamless splicing, or for purposes other than splicing.

K.3 Decoder behaviour on splices

K.3.1 On non-seamless splices

As described above, a non-seamless splice is a splice which results in a decoding discontinuity.

It shall be noted that with such a splice, the constraints related to the decoding discontinuity (see 2.7.6) shall be fulfilled. In particular:

- a PTS shall be encoded for the first access unit of the 'new' stream (except during trick mode operation or when low_delay = '1');
- the decoding time derived from this PTS (or from the associated DTS) shall not be earlier than the seamless decoding time;
- in the case of a video elementary stream, if the splicing packet does not end on a sequence_end_code(), the 'new' stream shall begin with a sequence_end_code() immediately followed by a sequence_header().

In theory, since they introduce decoding discontinuities, such splices result in a non-continuous presentation of presentation units (i.e., a variable length dead time between the display of two consecutive pictures, or between two consecutive audio frames). In practice, the result will depend on how the decoder is implemented, especially in video. With some video decoders, the freezing of one or more pictures may be the preferred solution. See Part 4 of ISO/IEC 13818.

K.3.2 On seamless splices

The aim of having no decoding discontinuity is to allow having no presentation discontinuity. In the case of audio, this can always be ensured. But it has to be noted that in the case of video, presentation continuity is in theory not possible in cases 1) and 2) below:

1) The 'old' stream ends on the end of a low-delay sequence, and the 'new' stream begins with the start of a non-low-delay sequence.
2) The 'new' stream ends on the end of a non-low-delay sequence, and the 'new' stream begins with the start of a low-delay sequence.

The effects induced by such situations is implementation dependent. For instance, in case 1, a picture may have to be presented during two frame periods, and in case 2, a picture may have to be skipped. However, it is technically possible that some implementations support such situations without any undesirable effect.

In addition, referring to 6.1.1.6 of Rec. ITU-T H.262 | ISO/IEC 13818-2, a sequence_end_code() shall be present before the first sequence_header() of the 'new' stream, if at least one sequence parameter (i.e., a parameter defined in the sequence header or in a sequence header extension) has a different value in both streams, with the only exception of those defining the quantization matrix. As an example, if the bit rate field has not the same value in the 'new' stream as in the 'old' one, a sequence_end_code() shall be present. Thus, if the splicing packet does not end on a sequence_end_code, the 'new' stream shall begin with a sequence_end_code followed by a sequence_header.

According to the previous paragraph, a sequence_end_code will be mandatory in most splices, even seamless ones. It has to be noted that Rec. ITU-T H.262 | ISO/IEC 13818-2 specifies the decoding process of video sequences (i.e., data comprised between a sequence_header() and a sequence_end_code()), and nothing is specified about how to handle a sequence change. Thus, for the behaviour of the decoders when such splices are encountered, refer to Part 4 of ISO/IEC 13818.

K.3.3 Buffer overflow

Even if both elementary streams obey the T-STD model before being spliced, it is not necessarily ensured that the STD buffers do not overflow with the spliced stream in the time interval during which bits of both streams are in these buffers.

In the case of constant bit rate video, if no particular conditions have been applied to the 'old' stream, and if no particular precautions have been taken during splicing, this overflow is possible in the case where the video bit rate of the 'new' stream is greater than the video bit rate of the 'old' one. Indeed, it is certainly true that the buffers MB_n and EB_n

of the T-STD do not overflow if bits are delivered to the T-STD at the 'old' rate. But if the delivery rate is switched to a higher value at the input of TB_n before 'old' bits are completely removed from the T-STD, the fullness of the STD buffers will become higher than if the 'old' stream had continued without splicing, and may cause overflow of EB_n and/or MB_n. In the case of variable bit rate video, the same problem can occur if the delivery rate of the 'new' stream is higher than the one for which provision was made during the creation of the 'old' stream. Such a situation is forbidden.

However, it is possible for the encoder generating the 'old' stream to add conditions in the VBV buffer management in the neighbourhood of splicing points, so that provision is made for any 'new' video bit rate lower than a chosen value. For instance, in the case of a seamless splicing point, such additional conditions can be indicated by a 'splice_type' value to which entries correspond in Table 2-7 through Table 2-20 for 'splice_decoding_delay' and 'max_splice_rate'. In that case, if the video bit rate of the 'new' stream is lower than 'max_splice_rate', it is ensured that the spliced stream will not lead to overflow during the time interval during which bits of both streams are in the T-STD buffer.

In the case where no such constraints have been applied, this problem can be avoided by introducing a dead time in the delivery of bits between the 'old' stream and the 'new' one, in order to let the T-STD buffers get sufficiently empty before the bits of the 'new' stream are delivered. If we call t_{in} the time at which the last byte of the last access unit of the 'old' stream enters the STD, and t_{out} the time at which it exits the STD, it is sufficient to ensure that no more bits enter the T-TD the time interval $[t_{in}, t_{out}]$ with the spliced stream than if the 'old' stream had continued without splicing. As an example, in the case where the 'old' stream has a constant bit rate R_{old}, and the 'new' one a constant bit rate R_{new}, it is sufficient to introduce a dead time T_d satisfying the following relations to avoid this risk of overflow:

$$T_d \geq 0 \text{ and } T_d \geq (t_{out} - t_{in}) \times (1 - R_{old}/R_{new})$$

Annex L

Registration procedure (see 2.9)

(This annex does not form an integral part of this Recommendation | International Standard.)

L.1 Procedure for the request of a Registered Identifier (RID)

Requesters of a RID shall apply to the Registration Authority. Registration forms shall be available from the Registration Authority. Information which the requester shall provide is given in L.3. Companies and organizations are eligible to apply.

L.2 Responsibilities of the Registration Authority

The primary responsibilities of the Registration Authority administrating the registration of copyright_identifiers is outlined in this subclause; certain other responsibilities may be found in the JTC 1 Directives. The Registration Authority shall:

a) implement a registration procedure for application for a unique RID in accordance with Annex H/JTC 1 Directives;

b) receive and process the applications for allocation of the work type code identifier from Copyright Registration Authority;

c) ascertain which applications received are in accordance with this registration procedure, and to inform the requester within 30 days of receipt of the application of their assigned RID;

d) inform application providers whose request is denied in writing within 30 days of receipt of the application, and also inform the requesting party of the appeals process;

e) maintain an accurate register of the allocated RID. Revisions to the contact information and technical specifications shall be accepted and maintained by the Registration Authority;

f) make the contents of this register available upon request to any interested party;

g) maintain a database of RID request forms, granted and denied. Parties seeking technical information on the format of private data which has a copyright_identifier shall have access to such information which is part of the database maintained by the Registration Authority;

h) report its activities to JTC 1, the ITTF and the JTC 1/SC 29 Secretariat, or their respective assignees, annually on a schedule mutually agreed upon.

L.2.1 Contact information of the Registration Authority

Organization Name:

Address:

Telephone:

Fax:

L.3 Responsibilities of parties requesting an RID

The party requesting an RID for the purpose of copyright identification shall:

a) apply using the form and procedures supplied by the Registration Authority;

b) provide contact information describing how a complete description of the copyright organization can be obtained on a non-discriminatory basis;

c) include technical details of the syntax and semantics of the data format used to describe the audiovisual works or other copyrighted works within the **additional_copyright_info** field. Once registered, the syntax used for the additional copyright information shall not change;

d) agree to institute the intended use of the granted copyright_identifier within a reasonable time-frame;

e) maintain a permanent record of the application form and the notification received from the Registration Authority of each granted copyright_identifier.

L.4 Appeal procedure for denied applications

The registration management group is formed to have jurisdiction over appeals relating to a denied request for an RID. The RMG shall have a membership who are nominated by P and L members of the ISO technical body responsible for this Recommendation | International Standard. It shall have a convenor and secretariat nominated from its members. The Registration Authority is entitled to nominate one non-voting observing member.

The responsibilities of the RMG shall be:

a) to review and act on all appeals within a reasonable time-frame;

b) to inform, in writing, organizations which make an appeal for reconsideration of its petition of the RMG's disposition of the matter;

c) to review the annual report of the Registration Authority summary of activities;

d) to supply ISO member bodies with information concerning the scope of operation of the Registration Authority.

Annex M

Registration application form (see 2.9)

(This annex does not form an integral part of this Recommendation | International Standard.)

M.1 Contact information of organization requesting a Registered Identifier (RID)

Organization Name:

Address:

Telephone:

Fax:

email:

M.2 Statement of an intention to apply the assigned RID

RID application domain: using guidelines to be provided by the Registration Authority.

M.3 Date of intended implementation of the RID

M.4 Authorized representative

Name:

Title:

Address:

Signature: ___________________________________

M.5 For official use only of the Registration Authority

Registration rejected: _______ Reason for rejection of the application: Registration granted: ________ Registration value: _______

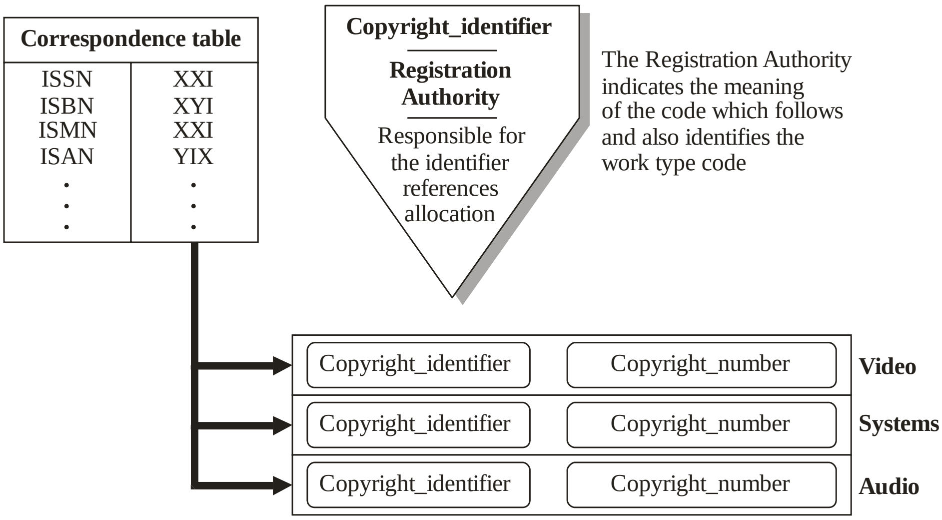
Attachment 1 – Attachment of technical details of the registered data format.

Attachment 2 – Attachment of notification of appeal procedure for rejected applications.

Annex N

Registration Authority
Diagram of administration structure (see 2.9)

(This annex does not form an integral part of this Recommendation | International Standard.)



H.222.0(12)_FN.1

Examples

Copyright_identifier	Copyright_number
I.S.B.N (for books)	2-11-0725-575 (ISBN number)
I.S.A.N (for audiovisual works)	1234567890123456 (ISAN number)

H.222.0(12)_FN.2

All the copyright_identifiers are registered by the Registration Authority, uniquely for copyright_numbers standardized by ISO. Each organization which allocates copyright_numbers, requests a specific copyright_identifier from the Registration Authority, e.g., Staatsbibliothek Preussischer Kulturbesitz, designated by ISO to manage I.S.B.N., asks for a specific copyright_identifier from the R.A. for book numbering.

Annex O

Registration procedure (see 2.10)

(This annex does not form an integral part of this Recommendation | International Standard.)

O.1 Procedure for the request of an RID

Requesters of an RID shall apply to the Registration Authority. Registration forms shall be available from the Registration Authority. The requester shall provide the information specified in O.4. Companies and organizations are eligible to apply.

O.2 Responsibilities of the Registration Authority

The primary responsibilities of the Registration Authority administrating the registration of private data format_identifiers is outlined in this annex; certain other responsibilities may be found in the JTC 1 Directives. The Registration Authority shall:

a) implement a registration procedure for application for a unique RID in accordance with the JTC 1 Directives;

b) receive and process the applications for allocation of an identifier from application providers;

c) ascertain which applications received are in accordance with this registration procedure, and to inform the requester within 30 days of receipt of the application of their assigned RID;

d) inform application providers whose request is denied in writing within 30 days of receipt of the application, and to consider resubmission of the application in a timely manner;

e) maintain an accurate register of the allocated identifiers. Revisions to format specifications shall be accepted and maintained by the Registration Authority;

f) make the contents of this register available upon request to National Bodies of JTC 1 that are members of ISO or IEC, to liaison organizations of ISO or IEC and to any interested party;

g) maintain a database of RID request forms, granted and denied. Parties seeking technical information on the format of private data which has an RID shall have access to such information which is part of the database maintained by the Registration Authority;

h) report its activities to JTC 1, the ITTF, and the SC 29 Secretariat, or their respective designees, annually;

i) accommodate the use of existing RIDs whenever possible.

O.3 Contact information for the Registration Authority

O.4 Responsibilities of parties requesting an RID

The party requesting a format_identifier shall:

a) apply, using the form and procedures supplied by the Registration Authority;

b) include a description of the purpose of the registered bit stream, and the required technical details as specified in the application form;

c) provide contact information describing how a complete description can be obtained on a non-discriminatory basis;

d) agree to institute the intended use of the granted RID within a reasonable time-frame;

e) to maintain a permanent record of the application form and the notification received from the Registration Authority of a granted RID.

O.5 Appeal procedure for denied applications

The Registration Management Group is formed to have jurisdiction over appeals to denied requests for an RID. The RMG shall have a membership who is nominated by P- and L-members of the ISO technical committee responsible for this Specification. It shall have a convenor and secretariat nominated from its members. The Registration Authority is entitled to nominate one non-voting observing member.

The responsibilities of the RMG shall be:

a) to review and act on all appeals within a reasonable time-frame;

b) to inform, in writing, organizations which make an appeal for reconsideration of its petition of the RMG's disposition of the matter;

c) to review the annual report of the Registration Authorities summary of activities;

d) to supply member bodies of ISO and National Committees of IEC with information concerning the scope of operation of the Registration Authority.

Annex P

Registration application form

(This annex does not form an integral part of this Recommendation | International Standard.)

P.1 Contact information of organization requesting an RID

Organization Name:

Address:

Telephone:

Fax:

email:

Telex:

P.2 Request for a specific RID

NOTE – If the system has already been implemented and is in use, fill in this item and also the item P.3 and then skip to P.6; otherwise leave this space blank and skip to P.4.

P.3 Short description of RID that is in use and date system that was implemented

P.4 Statement of an intention to apply the assigned RID

P.5 Date of intended implementation of the RID

P.6 Authorized representative

Name:

Title:

Address:

Signature: ___________________________________

P.7 For official use of the Registration Authority

Registration rejected: _______ Reason for rejection of the application: Registration granted: ________ Registration value: _______

Attachment 1 – Attachment of technical details of the registered data format.

Attachment 2 – Attachment of notification of appeal procedure for rejected applications.

Annex Q

T-STD and P-STD buffer models for ISO/IEC 13818-7 ADTS

(This annex does not form an integral part of this Recommendation | International Standard.)

Q.1 Introduction

The transport stream system target decoder model for audio streams is defined in 2.4.2. In this annex, the buffer model for ISO/IEC 13818-7 ADTS is described.

ISO/IEC 13818-7 ADTS audio streams can be recognized in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 multiplex through the presence of stream_id='110y yyyy' ('y' = "do not care") and stream_type=0x0F as defined in Tables 2-22 and 2-34.

Q.2 Leak rate from transport buffer

For audio except ISO/IEC 13818-7 ADTS, the leak rate from Transport Buffer is 2 Mbit/s. This rate is, however, lower than the maximum rate of ISO/IEC 13818-7 ADTS. Therefore, the leak rate for ISO/IEC 13818-7 ADTS streams is set to a different value from ISO/IEC 11172-3 and ISO/IEC 13818-3 audio streams.

ISO/IEC 13818-7 ADTS elementary stream consists of one or more channels. The maximum rate of each channel is 576 kbit/s where the sampling frequency is 96 kHz. Therefore, the leak rate for ISO/IEC 13818-7 ADTS is calculated as in the following equation.

$$Rx_n = 1.2 \times R_{max} \times N \text{ bits per second}$$

where:

R_{max} is a constant 576 kbit/s as defined in 3.2.2 of ISO/IEC 13818-7. It is an upper bound of the bit rate per channel of AAC ADTS stream corresponding to the maximum value of sampling frequency (i.e., Fs = 96 kHz),

and where:

N is the number of audio channels that require their own decoder buffer in this elementary stream (i.e., individual channel streams in a single channel element or channel pair element and independently switched coupling channel elements).

Q.3 Buffer size

For audio except ISO/IEC 13818-7 ADTS, the main buffer size is 3584 bytes. This size is, however, smaller than the maximum decoder input buffer size of ISO/IEC 13818-7 ADTS. Therefore, the main buffer size for the ISO/IEC 13818-7 ADTS stream is set to a different value from ISO/IEC 11172-3 and ISO/IEC 13818-3 audio streams.

The main buffer size for ISO/IEC 13818-7 ADTS is calculated as follows:

$$BS_n = BS_{mux} + BS_{dec} + BS_{oh}$$

where BS_{oh}, PES packet overhead buffering is defined as:

$$BS_{oh} = 528 \text{ bytes}$$

and BS_{mux}, additional multiplexing buffering is defined as:

$$BS_{mux} = 0.004 \text{ seconds} \times R_{max} \times N$$

and BS_{dec}, access unit buffering is defined as:

$BS_{dec} = 6144 \text{ bits} \times N$

where:

R_{max} is a constant 576 kbit/s as defined in 3.2.2 of ISO/IEC 13818-7. It is an upper bound of the bit rate per channel of AAC ADTS stream corresponding to the maximum value of sampling frequency (i.e., Fs = 96 kHz),

and where:

N is the number of audio channels that require their own decoder buffer in this elementary stream (i.e., individual channel streams in a single channel element or channel pair element and independently switched coupling channel elements).

Q.3.1 TBS_n: same as other audio

In term of the smoothing buffer, there is no difference in TB_n between ISO/IEC 13818-7 ADTS and other audio streams. Consequently, it is not necessary to change TBS_n, which is size of TB_n.

Q.3.2 BS_{mux}: different from other audio

BS_{mux}, additional multiplexing buffering, shall be changed to accept up to 4 ms of delay jitter. This is similar to the approach taken for other streams in Rec. ITU-T H.222.0 | ISO/IEC 13818-1.

Q.3.3 BS_{dec}: different from other audio

BS_{dec}, access unit buffering is based on the decoder input buffer size of the elementary stream. As defined in 3.2.2 of ISO/IEC 13818-7, total decoder input buffer size is 6144 bits multiplied by the number of channels which require their each decoder input buffer.

Q.3.4 BS_{oh}: different from other audio

BS_{oh} corresponds to the PES packet header overhead.

In 2.4.2.7,

> *The delay of any data through the System Target Decoders buffers shall be less than or equal to one second except for still picture video data.*

Besides, in 2.7.4,

> *The program stream and transport stream shall be constructed so that the maximum difference between coded presentation time stamp referring to each elementary video or audio stream is 0.7 seconds.*

BS_{oh} shall be set to the appropriate size corresponding to the PES packet header overhead when AAC stream is packetized with the above rules. The maximum size of PES packet header is 264 bytes. Therefore, BS_{oh} = 528 bytes, i.e., twice the maximum size of PES packet header, assures that at least two PES packet headers can enter the main buffer regardless of the size of PES packet header. It means that PES packet header with PTS can be inserted at less than 0.7 seconds intervals even when the data of one second will be in the main buffer.

Example: sampling frequency is 48 kHz

The size of PES packet header without any optional fields except PTS is 18 bytes. The number of Access Units of one second is about 47. When the data of one second is in the main buffer (i.e., the worst case), PES packet header overhead can fit the BS_{oh} with packetizing more than or equal to two Access Units into one packet.

number_of_AU = 48 kHz/1024 = 46 875 per second

(number_of_AU/2) × 18[byte] = 421 875 bytes < BS_{oh}

More frequent PES packet headers can fit BS_{oh}, if the delay of any data through the main buffer is shorter than one second.

Q.4 Conclusion

The decoder buffer model should cover the maximum size of buffer; however, AAC can handle up to 48 channels and very high bitrate. Therefore the 3 levels of number of channels, 2, 8 and 48, are used to define the leak rate and the main buffer size. In a case of 2, the same leak rate and main buffer size as the conventional values are used to keep the compatibility. In other cases (8 and 48), the proposed formulas are applied.

T-STD leak rate for ISO/IEC 13818-7 ADTS audio,

Number of channels	Rx_n [bit/s]
1 .. 2	2 000 000
3 .. 8	5 529 600
9 .. 12	8 294 400
13 .. 48	33 177 600

Channels: The number of full-bandwidth audio output channels plus the number of independently switched coupling channel elements within the same elementary audio stream. For example, in the typical case that there are no independently switched coupling channel elements, mono is 1 channel, stereo is 2 channels and 5.1 channel surround is 5 channels (the LFE channel is not counted).

T-STD main buffer size or ISO/IEC 13818-7 ADTS audio

Number of channels	BS_n [bytes]
1 .. 2	3 584
3 .. 8	8 976
9 .. 12	12 804
13 .. 48	51 216

Channels: The number of full-bandwidth audio output channels plus the number of independently switched coupling channel elements within the same elementary audio stream. For example, in the typical case that there are no independently switched coupling channel elements, mono is 1 channel, stereo is 2 channels and 5.1 channel surround is 5 channels (the LFE channel is not counted).

For program stream, the above main buffer size should be set in the P-STD_buffer_scale and P-STD_buffer_size as follows.

Number of Channels	P-STD_buffer_scale	P-STD_buffer_size
1 .. 2	0	28
3 .. 8	0	71
9 .. 48	0	401

Annex R

Carriage of ISO/IEC 14496 scenes in Rec. ITU-T H.222.0 | ISO/IEC 13818-1

(This annex does not form an integral part of this Recommendation | International Standard.)

R.1 Content access procedure for ISO/IEC 14496 program components within a program stream

The following provides a reference receiver acquisition procedure for accessing ISO/IEC 14496 program elements in a Rec. ITU-T H.222.0 | ISO/IEC 13818-1 program stream. Here, it is assumed that the program stream includes a Program Stream Map (conveyed in a PES packet having stream_id equal to 0xBC):

- Acquire the Program Stream Map.
- Identify the IOD descriptor in the first descriptor loop.
- Identify the ES_IDs of the object descriptor, scene description and other streams described within the initial object descriptor.
- Acquire the SL descriptor and FMC descriptor in the second descriptor loop for elementary_stream_ids 0xFA and 0xFB, as applicable.
- Generate from these descriptors a stream map table between ES_IDs and associated elementary_stream_id plus FlexMux channel, if applicable.
- Locate the object descriptor stream using its ES_ID and the stream map table.
- Locate other streams described in the initial object descriptor using their ES_ID and the stream map table.
- Continuously monitor the object descriptor stream and identify ES_IDs of additional streams.
- Locate the additional streams using their ES_ID and the stream map table.

Figure R.1 gives an example of ISO/IEC 14496 content in a program stream, consisting of object descriptor stream, scene description stream (BIFS-Command), BIFS-Anim stream and IPMP stream. All ISO/IEC 14496 streams are multiplexed in a single FlexMux stream.

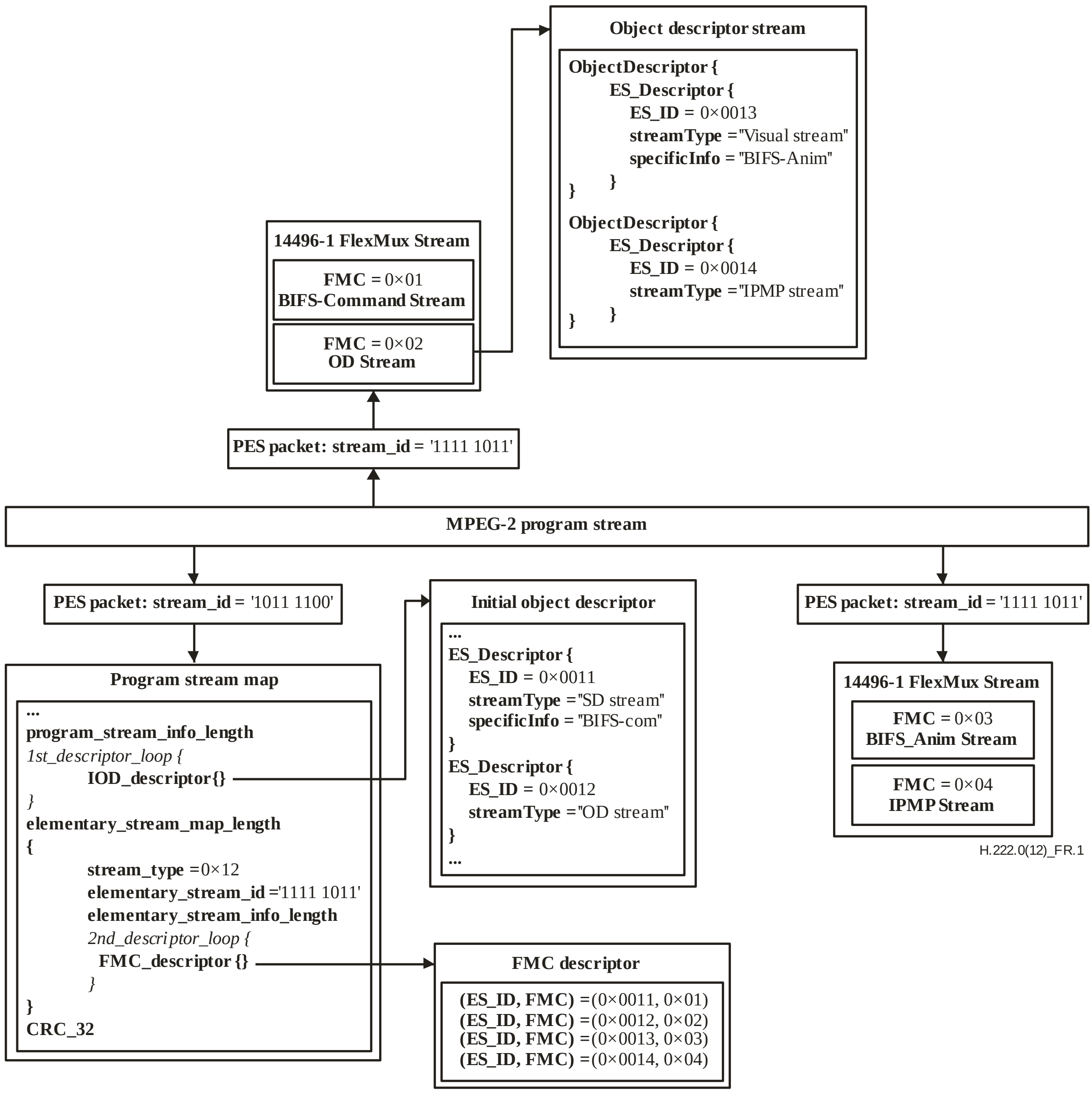


Figure R.1 – Example of ISO/IEC 14496 content in a program stream

R.2 Content access procedure for ISO/IEC 14496 program components within a transport stream

The following provides a reference receiver acquisition procedure for accessing ISO/IEC 14496 program elements in an Rec. ITU-T H.222.0 | ISO/IEC 13818-1 transport stream:

- Acquire the Program Map Table for the desired program.
- Identify the IOD descriptor in the first descriptor loop.
- Identify the ES_IDs of the object descriptor, scene description and other streams described within the initial object descriptor.
- Acquire the set of all SL descriptors and FMC descriptors present in the second descriptor loop for any of the elementary_PIDs.
- Generate from these descriptors a stream map table between ES_IDs and associated elementary_PID plus FlexMux channel, if applicable.
- Locate the object descriptor stream using its ES_ID and the stream map table.
- Locate other streams described in the initial object descriptor using their ES_ID and the stream map table.

- Continuously monitor the object descriptor stream and identify ES_IDs of additional streams.
- Locate the additional streams using their ES_ID and the stream map table.
- If ODUpdate_descriptors are present in the first descriptor loop, process the ObjectDescriptor_Update as defined in 2.6.92.

Figure R.2 gives an example of ISO/IEC 14496 program elements in a transport stream, consisting of object descriptor stream, scene description stream (BIFS-Command), BIFS-Anim and IPMP elementary streams. BIFS-Command and OD stream are conveyed by means of ISO_IEC_14496_sections, while BIFS-Anim and IPMP elementary streams are conveyed in PES packets referenced by two distinct elementary_PID values, without the use of the ISO/IEC 14496-1 FlexMux tool.

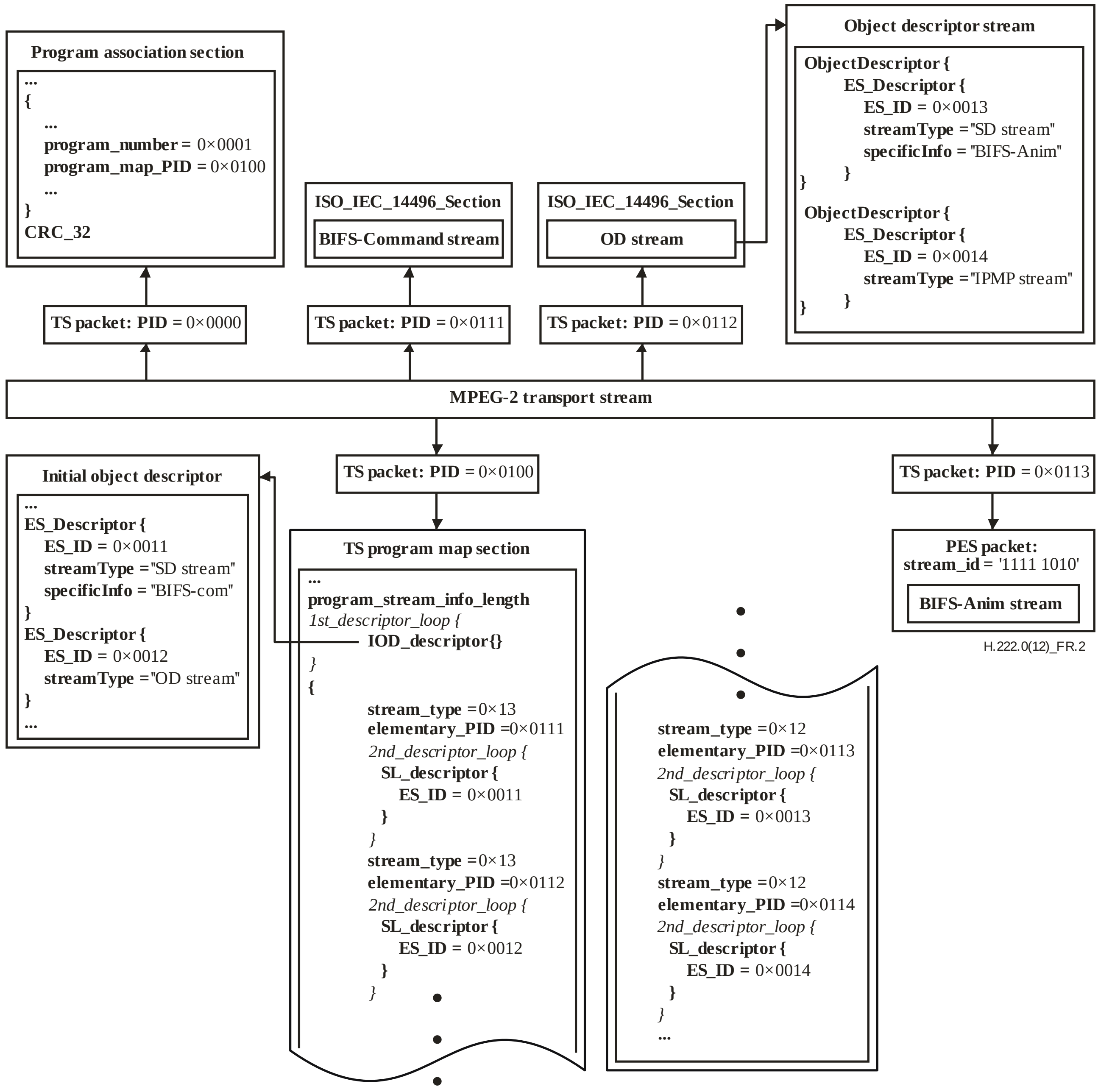


Figure R.2 – Example of ISO/IEC 14496 content in a transport stream

Figure R.3 gives an example of ISO/IEC 14496 program elements in a transport stream, consisting of a scene description stream (BIFS-Command), and audio and video streams natively carried over PES (no SL packetization or FlexMux). BIFS-Command stream are conveyed by means of ISO_IEC_14496_sections, and the BIFS scene directly refers to the audio and video streams in the transport stream through "pid://" URLs in the BIFS media nodes.

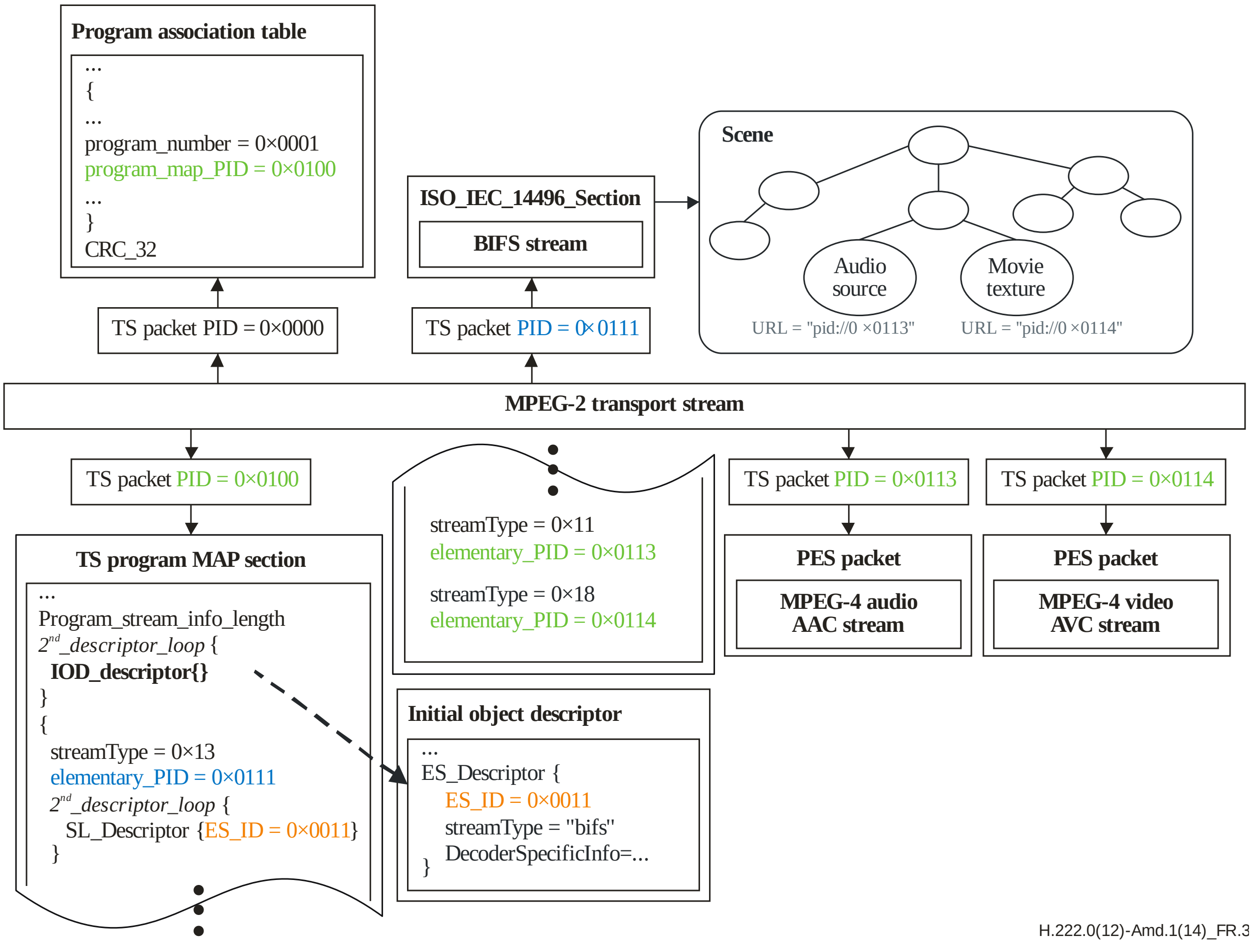


Figure R.3 – Usage of MPEG-4 in a transport stream with BIFS scene referring to native PES

Figure R.4 gives an example of ISO/IEC 14496 program elements in a transport stream, consisting of a scene description stream (BIFS-Command) and an image stream. The BIFS-Command stream is conveyed by means of ISO_IEC_14496_sections, the image stream is conveyed in PES packets using SL packetization. The ObjectDescriptor associated with the image stream is conveyed by means of an ODUpdate_Descriptor in the first descriptor loop of the PMT.

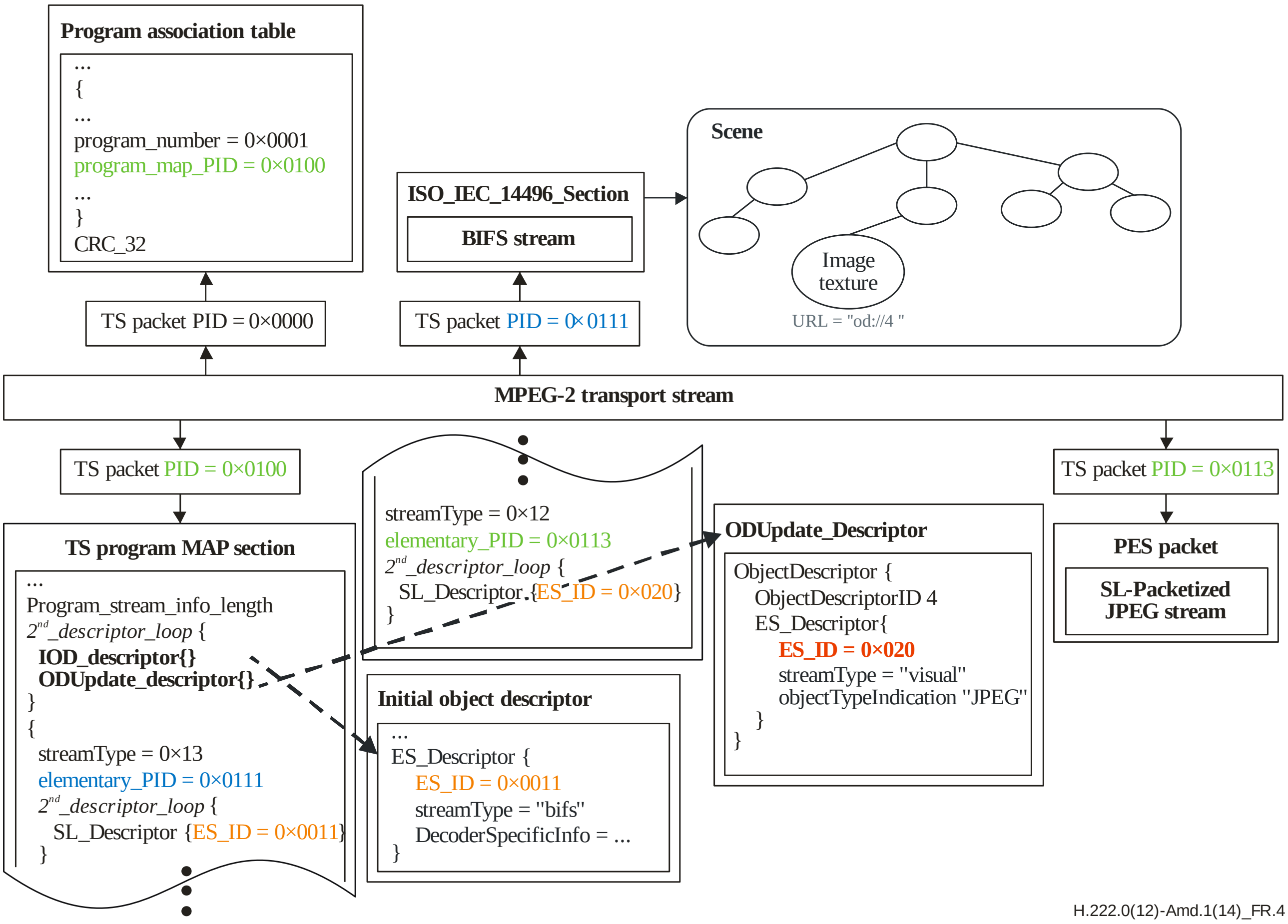


Figure R.4 – Usage of MPEG-4 in a transport stream with an ODUpdate_descriptor carrying an image ObjectDescriptor in the PMT

Annex S

Carriage of JPEG 2000 part 1 video over MPEG-2 transport streams

(This annex forms an integral part of this Recommendation | International Standard.)

S.1 Introduction

This annex specifies normative constraints for the carriage of JPEG 2000 video in an MPEG-2 transport stream. The parameters specified include mapping of J2K video streams into MPEG-2 transport packets, signalling of J2K video streams as well as T-STD parameters for various profiles. Transport of J2K video shall be limited to transport stream only. Program stream support may be added in the future based on application requirements.

S.2 J2K video access unit, J2K video elementary stream, J2K video sequence and J2K still picture

The J2K video access unit contains the elementary stream (elsm) header created in a manner following Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 concatenated with self-contained Rec. ITU-T T.800 | ISO/IEC 15444-1 codestream(s). The (elsm) header contains all video-related parameters necessary to display the decoded codestream(s). Multiple codestreams may comprise an access unit when, for instance, the access unit is an interlaced frame. The J2K video elementary stream is a progression of J2K access units and the J2K video sequence is a subset of J2K video elementary stream where all the J2K access units have the same parameters in the (elsm) header.

The J2K still picture (system) consists of a J2K video sequence which contains exactly one J2K access unit. This still picture has an associated PTS and the presentation time of succeeding pictures, if any, is later than that of the still picture by at least two picture periods. The J2K still picture (system) mode is used to support transmission of J2K video access units at a rate much lower than the display frame rate (determined by the difference in PTS values between successive J2K access units). J2K still picture can be used in applications such as 'slide show' and 'stills with Music'.

S.3 Elementary stream header (elsm) and mapping to PES packets

Table S.1 shows each portion of the elsm header as defined in Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 and the concatenation of detailed metadata boxes within the elsm header.

Table S.1 – J2K Access unit elementary stream header

Syntax	No. of bits	Mnemonic
j2k_es() {		
j2k_elsm '0x656c 736d'	32	bslbf
// j2k_frat		
frat_box_code '0x6672 6174'	32	bslbf
frat_denominator	16	uimsbf
frat_numerator	16	uimsbf
// j2k_brat		
brat_box_code = '0x6272 6174'	32	bslbf
Maxbr	32	uimsbf
Auf1	32	uimsbf
// If (interlaced_video == 1) {		
Auf2	32	uimsbf
}		
// j2k_fiel		
// If (interlaced_video == 1) {		
fiel_box_code = 0x6669 656c	32	bslbf
fic	8	uimsbf
fio	8	uimsbf
}		
// j2k_tcod		
tocd_code = 0x7463 6f64	32	bslbf
HH (0-23)	8	uimsbf
MM (0-59)	8	uimsbf
SS (0-59)	8	uimsbf
FF (1-60)	8	uimsbf
// j2k_bcol		
bcol_code = 0x6263 686c	32	bslbf
bcol_colcr	8	uimsbf
reserved	8	bsfbf
}		

j2k_frat – A field box defined in Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 corresponding to the required frat box containing frame rate coding.

j2k_brat – A field box defined in Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 corresponding to the required brat box containing the maximum instantaneous bit rate of the elementary stream.

j2k_fiel – A field box defined in Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 corresponding to the optional fiel box containing interleaved field coding. If the j2k_fiel is present, there shall be two contiguous codestreams present.

j2k_tcod – A field box defined in Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 corresponding to the required tcod box containing time code and frame count information of the J2K video access unit.

j2k_bcol – A field box defined in Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 corresponding to the required bcol box containing broadcast colour specification coding.

Figure S.1 shows the structure and mapping of J2K video access unit into PES packets. JPEG 2000 represents each frame as one or two Part 1 (Rec. ITU-T T.800 | ISO/IEC 15444-1) contiguous codestreams. The codestream main header, included within each contiguous codestream, contains all information to decode its image, including the image size and the profile indicator, called a SIZ marker in Rec. ITU-T T.800 | ISO/IEC 15444-1. In addition to each codestream's main header, Annex M of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 adds an elementary stream (elsm) header containing video-related information, as shown in Table S.1 and the j2k_video_descriptor in 2.6.80.

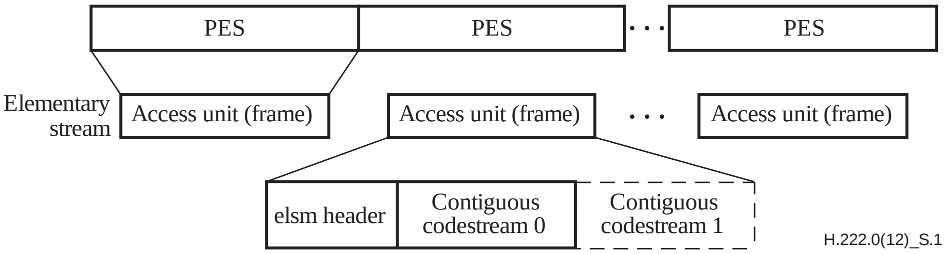


Figure S.1 – Structure and order of JPEG 2000 access units

S.4 J2K transport constraints

When a J2K video elementary stream conforming to one or more profiles as defined in Rec. ITU-T T.800 | ISO/IEC 15444-1 is transported using MPEG-2 systems, the following constraints apply:

1) Each J2K access unit shall contain an elementary stream header (elsm) defined in Table S.1 (as specified in Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3) followed by one or two codestream(s).

2) Each J2K codestream main header contains a SIZ marker segment that includes a RSIZ capability parameter. Both the SIZ marker segment and RSIZ capability parameter are defined in Annex A of Rec. ITU-T T.800 | ISO/IEC 15444-1 equating to the profile_and_level parameter (see 2.6.81).

3) The J2K video access units shall be ordered in the J2K video elementary stream in a monotonic display order.

4) Each J2K video access unit shall include a PES header with PTS and each PES packet shall contain exactly one J2K video access unit.

5) For successive J2K video access units, the increments to PTS shall be consistent with increments to corresponding J2K_tcod parameters in the elsm header.

6) The following constraints apply to the coding of syntax elements in the adaptation header for transport of J2K video elementary stream:

   a) Both random_access_indicator and elementary_stream_priority_indicator flags can be set to '1' for each J2K video access unit contained in the transport packet. Applications may limit the signalling of random access based on their use cases. Any J2K video access unit is also a J2K video 'access point' required for random access.

   b) All other flags should be set appropriately.

7) The following constraints apply to the coding of syntax elements in the PES header for transport of J2K video elementary stream:

   a) stream_id shall be set to 0xBD (same as Private_stream_1).

   b) PES_packet_length shall be set to 0x0000.

   c) data_alignment_indicator shall be set to '1'. Also see S.5 for alignment constraints.

   d) PTS_DTS_flags shall be set to '10'. This precludes display re-ordering for J2K video access units.

   e) All other flags should be set appropriately.

S.5 Interpretation of flags in adaptation and PES headers for J2K video elementary streams

The interpretation, extensions, use and constraints for the following syntax elements in the adaptation header (2.4.3.4 and 2.4.3.5) and PES header (2.4.3.6 and 2.4.3.7) for JPEG 2000 Part 1 video are defined in this clause:

- In the semantics for discontinuity_indicator (see 2.4.3.5), a J2K video elementary stream access point is defined as the first byte of the J2K video access unit.
- The elementary_stream_priority_indicator in adaptation header (see 2.4.3.4 and 2.4.3.5) may be set to '1' if the transport packet payload contains the start of a J2K video access unit.
- For J2K video elementary streams, when the data_alignment_indicator (see 2.4.3.6) is set to '1', the PES packet header shall be immediately followed by the first byte of J2K video access unit. The data_stream_alignment_descriptor is optional and, if included for J2K video, the alignment_type shall be set to 0x00.

For J2K video elementary streams conforming to one or more profiles defined in Rec. ITU-T T.800 | ISO/IEC 15444-1, if a PTS is present in the PES packet header (see 2.4.3.7), it shall refer to the first byte of J2K video access unit that commences in this PES packet.

S.6 T-STD extension for J2K video elementary streams

S.6.1 General

The T-STD model includes a transport buffer TB_n and J2K video elementary stream buffer EB_n for decoding of each J2K video elementary stream n. See Figure S.2.

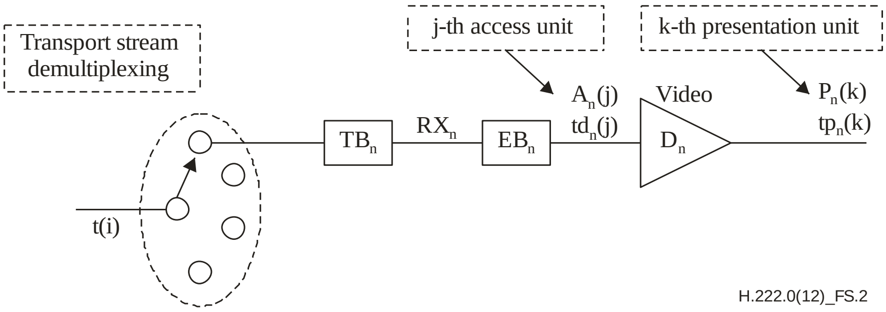


Figure S.2 – T-STD model extensions for J2K Video

The following notation is used to describe the transport stream system target decoder and is partially illustrated in Figure 2-1.

i	index to bytes in the transport stream. The first byte has index 0.
j	is an index to access units in the elementary streams.
k	index to presentation units in the elementary streams.
n	is an index to the elementary streams.
p	is an index to transport stream packets in the transport stream.
t(i)	indicates the time in seconds at which the i-th byte of the transport stream enters the system target decoder. The value t(0) is an arbitrary constant.
PCR(i)	is the time encoded in the PCR field measured in units of the period of the 27 MHz system clock where i is the byte index of the final byte of the program_clock_reference_base field.
$A_n(j)$	is the j-th access unit in elementary stream n. $A_n(j)$ is indexed in decoding order.
$td_n(j)$	is the decoding time, measured in seconds, in the system target decoder of the j-th access unit in elementary stream n.
$P_n(k)$	is the k-th presentation unit in elementary stream n. $P_n(k)$ results from decoding $A_n(j)$. $P_n(k)$ is indexed in presentation order.
$tp_n(k)$	is the presentation time, measured in seconds, in the system target decoder of the k-th presentation unit in elementary stream n.
t	is time measured in seconds.
$F_n(t)$	is the fullness, measured in bytes, of the system target decoder input buffer for elementary stream n at time t.
B_n	is the main buffer for elementary stream n. It is present only for audio elementary streams.
BS_n	is the size of buffer, B_n, measured in bytes.
EB_n	is the elementary stream buffer for elementary stream n. It is present only for video elementary streams.
EBS_n	is the size of the elementary stream buffer EB_n, measured in bytes.
TB_n	is the transport buffer for elementary stream n.
TBS_n	is the size of TB_n, measured in bytes.
Rx_n	is the rate at which data are removed from TB_n.

TB_n and EB_n buffer management

The input to buffer TB_n and its size TBS_n are specified in 2.4.2.4. The rate Rx_n between TB_n and buffer EB_n and the following constraints apply for the carriage of a J2K video elementary stream:

Size EBS_n of buffer EB_n: See Table S.2 for the maximum buffer size based on profile and level of J2K video stream.

Rate Rx_n:

when there is no data in TB_n then Rx_n is equal to zero,

otherwise: Rx_n = bit_rate,

where bit_rate is the bit rate as specified for the profile level in Table S.2. All J2K video payload data bytes enter EB_n instantaneously upon leaving TB_n.

Removal of J2K access units from EB_n

Each J2K video access unit $A_n(j)$ that is present in EB_n is removed instantaneously at time $td_n(j)$. The decoding time $td_n(j)$ is specified by the PTS (as PTS = DTS for J2K video elementary streams) in the PES header for the J2K video access unit.

STD delay

The total delay of any J2K video elementary stream data other than J2K still picture data through the system target decoders buffers TB_n and EB_n shall be constrained by $td_n(j) - t(i) \leq 1$ second for all j, and all bytes i in J2K video access unit $A_n(j)$.

The delay of any J2K still picture data through the system target decoders buffers TB_n and EB_n shall be constrained by $td_n(j) - t(i) \leq 60$ seconds for all j, and all bytes i in J2K video access unit $A_n(j)$.

Buffer management conditions

Transport streams shall be constructed so that the following conditions for buffer management are satisfied:

- TB_n shall not overflow and shall be empty at least once every second.
- EB_n shall not overflow or underflow.

NOTE – EB_n shall not underflow as J2K video elementary streams do not support low delay mode.

S.6.2 J2K video elementary stream buffer size

Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 elementary stream supplies parameters suitable for determining a standard decoder model buffer size. This CODEC uses variable rate compression. Profiles from Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3 specify operating levels that limit the maximum compressed bit rate of the Rec. ITU-T T.800 | ISO/IEC 15444-1 code stream. Moreover, within access units an elementary stream header contains the present maximum bit rate an encoder must support to decode video without overflow. A decoder may scale the buffer size by reading the maximum bit rate expected by a particular sequence of frames from the elsm header. This header information shall not exceed the limit set by the operating level specified in Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3.

Table S.2 – Operating levels and maximum buffer size for JPEG 2000 broadcast profiles (from Table A.48 in Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3)

Levels (Note)	Max. compressed bit rate (Mbit/s)	Maximum buffer size for a 60.0 Hz frame rate, progressive video (MBytes)
Level 1	200	1.25
Level 2	200	1.25
Level 3	200	1.25
Level 4	400	2.5
Level 5	800	5
Level 6	1600	10
Level 7	Unspecified	Unspecified
NOTE – Levels are specified in the JPEG 2000 codestream main header SIZ marker segment, RSIZ capability parameter. Refer to Table A.10 of Rec. ITU-T T.800 (2002)/Amd.3 | ISO/IEC 15444-1:2004/Amd.3.		

Annex T

MIME type for MPEG-2 transport streams

(This annex does not form an integral part of this Recommendation | International Standard.)

T.1 Introduction

This annex provides the formal MIME type registration for MPEG-2 transport streams. It is referenced from the registry at http://www.iana.org/.

T.2 MIME type and subtype

MIME media type name:
video

MIME subtype name:
mp2t

Required parameters:
none

Optional parameters:
The '`profiles`' parameter as documented in T.2.1

The '`codecs`' parameter as document in T.2.2

Encoding considerations:

This type is defined for general use; for transfer via RTP see IETF RFC 3550.

Security considerations:
see T.3

Interoperability considerations:

The specification defines a platform-independent expression of a presentation, and it is intended that wide interoperability can be achieved.

Published specification:
ITU-T H.222.0 | ISO/IEC 13818-1, Information technology – Generic coding of moving pictures and associated audio information: Systems

Applications that use this media type:
various, including video streaming and video broadcasting applications

Additional information:
File extension(s):

.ts

Intended usage:

COMMON

Other information/General comments:

none

Person to contact for further information:

Name:

David Singer

e-mail:

Singer@apple.com

Change controller:

ISO/IEC JTC1/SC29 (MPEG)

T.3 Security considerations

It is possible to inject non-compliant streams (audio, video and systems) in the transport stream to overload the receiver/decoder's buffers. This might compromise the functionality of the receiver or even crash it.

An MPEG-2 transport stream is an extensible container format, and hence might carry streams that have active aspects (e.g., contain script snippets). If those subsystems are not properly defined or implemented, it may be possible to crash the receiver or temporarily make it unavailable.

T.4 Parameters

T.4.1 The profiles parameter

Parameter Name: `profiles`

Parameter Value: The 'profiles' parameter is an optional parameter that indicates one or more profiles to which the stream claims conformance. The contents of this attribute shall conform to either the `pro-simple` or `pro-fancy` productions of IETF RFC 6381, Section 4.5. The profile identifiers reported in the MIME type parameter takes as value the `transport_profile`, coded as a decimal integer, e.g., `profiles="1"` for streams conforming to the 'complete' profile.

Example: `video/mp2t;profiles="1"`

T.4.2 The codecs parameter

Parameter Name: `codecs`

Parameter Value:

The 'codecs' parameter is an optional parameter that indicates one or more codecs which are used for the elementary streams in the MPEG-2 TS. The contents of this attribute shall conform to either the `pro-simple` or `pro-fancy` productions of IETF RFC 6381, Section 3.2. IETF RFC 6381 defines '`codecs`' parameter as a single value, or a comma-separated list of values identifying the codec(s), where each value consists of one or more dot-separated elements. The first element of such a value can be derived from the value of `stream_type` in the `program_map_section`. If the `stream_type` value appears in Table T.1, the value in the `first element` column shall be used as the first element. If the `stream_type` value is between `0x80` and `0xFF`, the first element may be derived in ways not specified by this annex (e.g., by examining the contents of `registration_descriptor`, if used).

Table T.1 – 'codecs' parameter values for some specific stream_type values

stream_type	first element
0x01	mp1v
0x02	mp2v
0x03	mp1a
0x04	mp2a
0x0F	mp2a
0x10	mp4v
0x11	mp4a
0x1B	avc1
0x1C	mp4a
0x1D	tx3g
0x1F	svc1
0x20	mvc1
0x21	mjp2
0x2F	vqme

Wherever a definition for additional elements exists in IETF RFC 6381 for a given first element, the definition in IETF RFC 6381 shall be followed.

When the first element of a value is 'mp1a', 'mp1v','mp2a' or 'mp2v', the second element of the codecs parameter value is the hexadecimal representation of the MP4 Registration Authority ObjectTypeIndication (OTI) for the appropriate specification and profile.

When the first element of a value is 'mp1a.6B' (ISO/IEC 11172-3), or 'mp2a.069' (i.e., ISO/IEC 13818-3), the third element of the codecs parameter value is the hexadecimal representation of the 2-bit layer, as defined in 2.6.4.

Examples:

ISO/IEC 13818-2 Main Profile

`video/mp2ts;codecs="mp2v.61"`

ISO/IEC 11172-3 layer 3 is represented

`video/mp2ts;codecs="mp2a.6B.03"`

ISO/IEC 13818-3 layer 2 is represented

`video/mp2ts;codecs="mp2a.69"`

ISO/IEC 13818-7 Low Complexity Profile

`video/mp2ts;codecs="mp2a.67"`

Dolby AC-3 audio (per ATSC A/52, AC-3 audio has stream_type 0x81 and format_identifier "AC-3" in the `registration_descriptor`)

`video/mp2ts;codecs="ac-3"`

ISO/IEC 13818-2 Main Profile Video together with ISO/IEC 13818-7 audio

`video/mp2ts;codecs="mp2v.61,mp2a.67"`

Annex U

Carriage of timeline and external media information over MPEG-2 transport streams

(This annex forms an integral part of this Recommendation | International Standard.)

U.1 Introduction

U.1.1 General

This annex specifies a format for carriage of timeline and location of external media resource that may be used as a synchronized enhancement of an MPEG-2 transport stream. The possible resolving, consumption and rendering of external media indicated in the stream are outside the scope of this Recommendation | International Standard.

The format specifies the mapping of the transport stream program clock to an embedded timeline, the signalling of associated external resources, hereafter called add-on(s), and the signalling of prefetching events. The format is designed to be compact in order to fit within one TS packet for common use cases. The mapping of the embedded timeline indicated in the PES packet payload or in the adaptation field descriptor with the PTS value of the PES header of the PES packet provides a stable timeline for media streams in the program, regardless of PCR discontinuities or other timestamps rewriting that may happen in the network.

In the context of this annex, the "timeline and external media information" is called TEMI.

TEMI data may be carried directly in the adaptation field of a media component or carried in a dedicated PES, called TEMI stream.

The TEMI stream describes external data and associated timing for the program in the MPEG-2 Transport Stream with which the TEMI stream is associated through the Program Map Table.

TEMI data carried in adaptation field of a media component describes external data and associated timing for the program in the MPEG-2 Transport Stream carrying this media component.

This annex also specifies a format for carriage of boundary and labelling descriptors that may be used to indicate a boundary type for seamless content splicing or switching in the applications of Ad insertion, cloud DVR recording and segmentation of adaptive bit rate streaming. The possible resolving and consumption of the boundary descriptor and labelling descriptor indicated in the stream are outside the scope of this Recommendation | International Standard.

U.1.2 Notation

This annex makes extensive use of variable-length, where field length is specified prior to the field itself. An additional short-hand notation is used to improve readability in these cases: length field names are referenced within the "number of bits" column of syntax tables. The alias name for the length field is provided in parenthesis in non-bold font at the same line as the length field, and the number of bits is given as a function of that field.

In the example in Table U.1, SFL is an alias for the `some_field_length_qwords` field. As the latter can have values of 0..3, `some_field` can have lengths of 0, 64, 128 and 192 bits. Stating that `some_field` is a 0-bit field implies that `some_field` is not present (in the example below this would result in `some_structure()` being a 1-byte structure).

Table U.1 – Variable field length notation example

Syntax	No. bits	Mnemonic
some_structure {		
some_field_length_qwords (SFL)	**2**	**uimsbf**
reserved	**6**	**bslbf**
some_field	**SFL*64**	**uismbf**
}		

The full notation of the same structure is given in Table U.1*bis*.

Table U.1*bis* – Table U.1 in equivalent full notation

Syntax	No. bits	Mnemonic
some_structure {		
some_field_length_qwords	**2**	**uimsbf**
reserved	**6**	**bslbf**
if (some_field_length_qwords > 0) {		
if (some_field_length_qwords == 1) {		
some_field	**64**	**uismbf**
}		
if (some_field_length_qwords == 2) {		
some_field	**128**	**uismbf**
}		
if (some_field_length_qwords == 3) {		
some_field	**192**	**uismbf**
}		
}		
}		

U.1.3 Annex references

– ANSI/SCTE 35 2016, Digital Program Insertion Cueing Message for Cable

U.2 TEMI access unit and TEMI elementary stream

The format of the TEMI access unit is defined in Table U.2. TEMI access units shall be carried as PES packets using private_stream_1 streamID and identified in the Program Map Table by the stream type 0x26. There shall be at most one TEMI elementary stream declared in the Program Map Table.

The payload of a TEMI PES packet is a single complete TEMI_AU, i.e., there shall be one and only one complete TEMI Access Unit in a TEMI PES packet.

The TEMI PES Packet header shall contain a PTS timestamp, whose value is used to match the current System Time Clock with the timeline value embedded in the TEMI packet payload, as defined in Table U.2.

TEMI data is made of one or several AF descriptors. These AF descriptors may be sent in different access units and at different rates. A TEMI_AU contains one or more AF descriptors. Since AF descriptors are independently decodable, all TEMI access units are random access points.

NOTE 1 – In order to avoid interpolation issues when frame-accurate synchronization is required, the indicated PTS should be the same as the PTS of the associated video or audio stream for which frame accurate sync is needed.

NOTE 2 – It is possible to perform timeline interpolation in-between TEMI access units, for example if multiple audio frames are packed in a single PES packet, or when the TEMI AU frequency is less than the media AU frequency. However, receivers detecting PCR discontinuities in-between TEMI AUs should be careful when performing interpolation.

Table U.2 – TEMI access unit

Syntax	No. of bits	Mnemonic
TEMI_AU {		
CRC_flag	**1**	**bslbf**
reserved	**7**	**bslbf**
for (i=0; i<N; i++) {		
af_descriptor()		
}		
if (CRC_flag) {		
CRC_32	**32**	**rpchof**
}		
}		

CRC_flag – A 1-bit flag, which when set to '1' indicates that a CRC field is present in the packet.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire payload of the TEMI access unit.

U.3 AF descriptors

U.3.1 Introduction

AF descriptors are structures used to carry various features of the timeline or other information. All AF descriptors have a format that begins with an 8-bit tag value. The tag value is followed by an 8-bit AF descriptor length and data fields. The following semantics apply to the descriptors defined throughout Annex U.

af_descr_tag – The af_descr_tag is an 8-bit field that identifies each AF descriptor.

Table U.3 provides the Rec. ITU-T H.222.0 | ISO/IEC 13818-1 defined, Rec. ITU-T H.222.0 | ISO/IEC 13818-1 reserved, and user available AF descriptor tag values.

af_descr_length – The af_descr_length is an 8-bit field specifying the number of bytes of the AF descriptor immediately following af_descr_length field.

Table U.3 – AF descriptor tags

AF descriptor tag	Identification
0x00 .. 0x03	Rec. ITU-T H.222.0	ISO/IEC 13818-1 reserved
0x04	Timeline descriptor
0x05	Location descriptor
0x06	BaseURL descriptor
0x07	Cets_byte_range_descriptor (see Note)
0x08	AF_MPEG-H_3dAudio_extStreamID_descriptor
0x09	AF_MPEG-H_3dAudio_multi-stream_descriptor
0x0A	AF_MPEG-H_3dAudio_command_descriptor
0x0B	Boundary Descriptor
0x0C	Labeling Descriptor
0x0D .. 0x7F	Rec. ITU-T H.222.0	ISO/IEC 13818-1 reserved
0x80 .. 0xFF	User Private
NOTE – See 6.4 in ISO/IEC 23001-9 (Common encryption of MPEG-2 transport streams) for description and usage.	

AF descriptors may be carried in the adaptation field of TS packets of a media elementary stream, as defined in 2.4.3.5.

U.3.2 Location descriptor

The location descriptor is used to signal the location of external data that can be synchronized with the program. It conveys several locations and their type (optionally including MIME types), along with the ability to signal upcoming external data association though a countdown until activation of the external data. It is possible to signal splicing of external data, by signalling that the newly associated data is temporary and the previous association will be re-used later on.

Table U.4 – TEMI location descriptor

Syntax	No. of bits	Mnemonic
temi_location_descriptor {		
af_descr_tag	**8**	**uimsbf**
af_descr_length	**8**	**uimsbf**
force_reload	**1**	**bslbf**
is_announcement	**1**	**bslbf**
splicing_flag	**1**	**bslbf**
use_base_temi_url	**1**	**bslbf**
reserved	**5**	**bslbf**
timeline_id	**7**	**uimsbf**
if (is_announcement = = '1') {		
timescale	**32**	**uimsbf**
time_before_activation	**32**	**uimsbf**
}		
if (use_base_temi_url = = '0') {		
url_scheme	**8**	**uimsbf**
url_path_length	**8**	**uimsbf**
for (i=0; i < url_path_length; i++) {		
url_path	**8**	**bslbf**
}		
}		
nb_addons	**8**	**uimsbf**
for (i=0; i < nb_addons; i++) {		
service_type	**8**	**uimsbf**
if (service_type = = 0) {		
mime_length	**8**	**uimsbf**
for (j=0; j < mime_length; j++) {		
mime_type	**8**	**bslbf**
}		
}		
url_subpath_len	**8**	**uimsbf**
for (j=0; j < url_subpath_len; j++) {		
addon_location	**8**	**bslbf**
}		
}		
}		

U.3.3 Semantic definition of fields in location descriptor

force_reload – when set to '1', indicates that the add-on descriptions received or fetched prior to receiving this descriptor may be obsolete and shall be reloaded before attempting to map media times or locate media components. Reloading may typically happen for manifest-based add-on such as MPEG-DASH or MPEG-MMT.

is_announcement – when set to '1', indicates that the add-on described by this descriptor is not yet active.

splicing_flag – when set to '1', indicates that the new add-on indicated by this descriptor temporarily interrupts the last defined add-on for which splicing_flag was not set. It is possible to have a sequence of add-ons with splicing_flag set. This allows terminal to optimize loading of the add-on when splicing period ends. There shall not be two temi_location_descriptor pointing to the same add-on with different values for splicing_flag, unless another temi_location_descriptor pointing to different add-ons is sent in-between with a splicing_flag set to '0'.

url_scheme – indicates the URL scheme to use for the URL. If url_scheme is not equal to 0, the scheme identified shall be prepended to the url_path, according to Table U.5.

Table U.5 – TEMI URL scheme types

TEMI URL scheme type	Scheme value
0	Scheme URL is included in url_path
1	"http://"
2	"https://"
3 .. 0x7F	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 reserved
0x80 .. 0xFF	User private

timeline_id – a unique identifier for this content location. If force_reload is set to '0' and another temi_location_descriptor with the same timeline_id and splicing_flag has already been received, the associated descriptions of the two descriptors shall be the same. If the splicing_flag differs for the same timeline_id, the timeline_id is reassigned to the new URL defined in the descriptor (i.e., redefinition of timeline_id).

timescale – indicates the timescale used to express the time_before_activation field in this message.

use_base_temi_url – when set to '1', indicates that the URL defined in the last received temi_base_url_descriptor shall be used as a base URL; when set to '0', a base URL is provided in the payload of this descriptor for the location described in this descriptor and only this descriptor.

time_before_activation – indicates the time in timescale units until the resource identified by addon_location becomes active; the ratio (time_before_activation / timescale) indicates a duration in seconds. An implementation may use this information to start prefetching content.

url_path_length – indicates the length in bytes of the base URL path; when set to 0, indicates an empty URL path.

url_path – base URL common to the different add-ons, if any; it shall be encoded without trailing zero character. This URL shall be a valid URL, as defined in clause 3 of RFC 3986, and may contain a Fragment and or a Query part.

nb_addons – indicates the number of add-ons that share this timeline. If set to 0, only one add-on is present at the location indicated by url_path, if this string is not empty. If url_path is empty and nb_addons is 0, this means that no service is associated with the current broadcast. If url_path is empty and nb_addons is not 0, url_subpath_len must be greater than 0.

service_type – indicates the type of add-on present at the given URL, as described in Table U-6. An implementation can decide to fetch or not the add-on based on this service type indication.

Table U.6 – TEMI service types

TEMI service type	Add-on type
0	Specified with MimeType
1	MPEG-DASH
2	ISO/IEC 14496-12 file
3	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Transport Stream
0x04 .. 0x7E	Rec. ITU-T H.222.0 | ISO/IEC 13818-1 Reserved
0x7F	Unknown service type
0x80 .. 0xFF	User Private

mime_type – indicates the mime type of the add-on available at the indicated location, as defined in RFC 2046. An implementation can decide to fetch or not the add-on based on this mime type indication.

url_subpath_length – indicates the length in bytes of the URL sub path; when set to 0, indicates an empty URL subpath.

url_subpath – indicates the URL sub path, without trailing zero character; this URL shall be a valid URL, as defined in clause 3 of RFC 3986. The URL for this add-on is obtained by merging url_subpath with the base URL path, as defined in clause 5 of RFC 3986.

U.3.4 Base URL descriptor

The base URL descriptor is used to assign a default base URL to all location descriptors (see Table U.7).

Table U.7 – TEMI base URL descriptor

Syntax	No. of bits	Mnemonic
temi_base_url_descriptor {		
af_descr_tag	**8**	**uimsbf**
af_descr_length	**8**	**uimsbf**
url_scheme	**8**	**uimsbf**
for (i=0; i < N; i++) {		
base_url_path	**8**	**bslbf**
}		
}		

U.3.5 Semantic definition of fields in Base URL descriptor

url_scheme – indicates the URL scheme to use for the URL. If url_scheme is not equal to 0, the scheme identified shall be prepended to the base_url_path, according to Table U.5.

base_url_path – base URL common to all following location descriptors, if any; it shall be encoded without trailing zero character. This URL shall be a valid path, as defined in clause 3 of RFC 3986, and may contain a fragment and or a query part.

U.3.6 Timeline descriptor

The timeline descriptor is used to carry timing information that can be used to synchronize external data (see Table U.8). When the descriptor is carried within a TEMI access unit, the included timing information is given for the PTS value of the TEMI access unit carrying the descriptor. When the descriptor is carried in the adaptation field of a media component, the included timing information is given for the PTS found in the PES header starting in the payload of this transport stream packet or in the first subsequent transport stream packet with payload_unit_start_indicator set to '1' on this component (same PID). This PES header shall have a PTS declared. For a given PES, there shall be at most one temi_timeline_descriptor with a timeline_id in the range [0,0x7F], a paused flag set to 0 and for which the last temi_location_descriptor received had an is_announcement flag set to 0. A temi_timeline_descriptor with a timeline_id in the range [0,0x7F], for which the last temi_location_descriptor received had an is_announcement flag set to '1', indicates the media time at which the timeline will start upon activation. This Recommendation | International Standard does not define any restrictions on the number of temi_timeline_descriptor using timeline_id values in the range [0x80, 0xFF] associated with a given PES.

In this section, this media PES packet is called the associated PES packet and the media PTS value is called the associated PTS.

Table U.8 – TEMI timeline descriptor

Syntax	No. of bits	Mnemonic
temi_timeline_descriptor {		
af_descr_tag	**8**	**uimsbf**
af_descr_length	**8**	**uimsbf**
has_timestamp	**2**	**uimsbf**
has_ntp	**1**	**bslbf**
has_ptp	**1**	**bslbf**
has_timecode	**2**	**uimsbf**
force_reload	**1**	**bslbf**
paused	**1**	**bslbf**
discontinuity	**1**	**bslbf**
reserved	**7**	**bslbf**
timeline_id	**8**	**uimsbf**
if (has_timestamp != 0) {		
timescale	**32**	**uimsbf**
if (has_timestamp = = 1) {		
media_timestamp	**32**	**uimsbf**
} else if (has_timestamp == 2) {		
media_timestamp	**64**	**uimsbf**
}		
}		
if (has_ntp = = '1') {		
ntp_timestamp	**64**	**uimsbf**
}		
if (has_ptp = = '1') {		
ptp_timestamp	**80**	**uimsbf**
}		
if (has_timecode != 0) {		
drop	**1**	**bslbf**
frames_per_tc_seconds	**15**	**uimsbf**
duration	**16**	**uimsbf**
if (has_timecode = = 1) {		
short_time_code	**24**	**uimsbf**
} else if (has_timecode = = 2) {		
long_time_code	**64**	**uimsbf**
}		
}		
}		

U.3.7 Semantic definition of fields in the timeline descriptor

has_timestamp – indicates a media timestamp will be carried in this descriptor, and indicates its type. Value 0 means no media timestamp is present, value 1 means a 32 bit media timestamp is present, value 2 means a 64 bit media timestamp is present, value 3 is reserved.

has_ntp – when set to '1', indicates that a NTP timestamp will be carried in this descriptor.

has_ptp – when set to '1', indicates that a PTP timestamp will be carried in this descriptor.

has_timecode – indicates that a frame timecode will be carried in this descriptor, and indicate its type. Value 0 means no frame timecode is present, value 1 means a compact frame timecode is present, value 2 means a full frame timecode is present, value 3 is reserved.

force_reload – when set to '1', indicates that the add-on descriptions received or fetched prior to receiving this descriptor may be obsolete and shall be reloaded before attempting to map media times or locate media components. Reloading typically happens for manifest-based add-on such as MPEG-DASH or MPEG-MMT.

paused – when set to '1', indicates that the timeline identified by timeline_id is currently paused; this typically happens when a timeline has to be paused but no splicing timeline is to be inserted during the pause. When a timeline whose timeline_id is in the range [0,0x7F] is running, all other timelines defined whose timeline_id are in the range [0, 0x7F] are implicitly in pause mode. For timelines whose timeline_id are in the range [0x80, 0xFF], this flag only indicates the running status of the timeline.

discontinuity – when set to '1', indicates that a discontinuity has happened in the timeline. If set to '0', no discontinuity happened since the last received temi_timeline_descriptor with the same value of timeline_id and same value of splicing_flag, if defined.

NOTE – An implementation may use this information to optimize playback of add-on content.

timeline_id – indicates the active timeline. timeline_id values in the range [0, 0x7F] are identified in a temi_location_descriptor; for such values of timeline_id, the content of this temi_timeline_descriptor shall be ignored if no temi_location_descriptor with the same timeline_id has been received. timeline_id values in the range [0x80, 0xFF] identify timelines defined by means beyond the scope of this Specification.

timescale – indicates the timescale used to express the media_timestamp field in this message; it indicates the amount by which the media time, as indicated by the media_timestamp field, increases within a 1 second interval when the timeline is not paused.

media_timestamp – indicates the media time in timescale units corresponding to the PES PTS value of this packet for the timeline identified by timeline_id. The value in this field is the media time modulo 2^N where N is the number of bits used to represent this field. The timeline may be interpolated between two temi_timeline_descriptor: let PTS_0 be the associated PTS of the temi_timeline_descriptor carrying media time MT_0; until a new temi_timeline_descriptor with the same timeline_id is received, the PTS of subsequent PES packets of other PIDs in this program is mapped to this TEMI timeline as follows:

$$MT_i = \text{timescale} * (PTS_i - PTS_0) / 90000 + MT_0$$

ntp_timestamp – A time value in 64-bit NTP timestamp format as defined in clause 6 of RFC 5905. The timeline may be interpolated between two temi_timeline_descriptor: let PTS_0 be the associated PTS of the temi_timeline_descriptor carrying NTP timestamp NTP_0; until a new temi_timeline_descriptor with the same timeline_id is received, the PTS of subsequent PES packets of other PIDs in this program is mapped to the NTP time NTP_i as follows (where NTP_0 and NTP_i are in units of seconds and fractions of a second):

$$NTP_i = (PTS_i - PTS_0) / 90000.0 + NTP_0$$

ptp_timestamp – A full 80 bits PTP timestamp as defined in IEEE 1588v2. The timeline may be interpolated between two temi_timeline_descriptor: let PTS_0 be the associated PTS of the temi_timeline_descriptor carrying PTP timestamp PTP_0; until a new temi_timeline_descriptor with the same timeline_id is received, the PTS of subsequent PES packets of other PIDs in this program is mapped to the PTP time PTP_i as follows (where PTP_0 and PTP_i are in units of seconds and fractions of a second):

$$PTP_i = (PTS_i - PTS_0) / 90000.0 + PTP_0$$

drop – drop-frame indication, as defined in clause 5 of RFC 5484.

frames_per_tc_second – the number of those frames that make a time-code second, as defined in clause 5 of RFC 5484.

duration – the duration in ticks of a frame expressed in the timescale of 90000 ticks per seconds, as defined in clause 5 of RFC5 484.

short_time_code – A compact 24-bit time code as defined clause 6.1 of RFC 5484.

long_time_code – A full 64-bit time code as defined clause 6.2 of RFC 5484.

short_time_code and **long_time_code** indicate the media time of the first access unit starting in the payload of the associated PES. Using the information of drop, duration and frames_per_tc_seconds, it is possible to interpolate the timing between two temi_timeline_descriptor.

U.3.8 MPEG-H 3dAudio extStreamID descriptor

The MPEG-H_3dAudio extStreamID descriptor (see Table U.9) is used to provide unique identification of streams. This descriptor shall be directly followed by either a location descriptor or a base URL descriptor which indicates the location of the external auxiliary stream.

The used auxiliaryStreamID in this descriptor value shall match the value of auxiliaryStreamID of the MPEG-H 3D audio multi-stream descriptor.

Table U.9 – TEMI MPEG-H_3dAudio_extStreamID descriptor

Syntax	No. of bits	Mnemonic
AF_MPEG-H_3dAudio_extStreamID_descriptor {		
af_descr_tag	**8**	**uimsbf**
af_descr_length	**8**	**uimsbf**
reserved	**1**	**bslbf**
auxiliaryStreamID	**7**	**uimsbf**
}		

U.3.9 Semantic definition of fields in AF_MPEG-H_3dAudio extStreamID descriptor

auxiliaryStreamID: provides a unique identifier for an external data stream that can be synchronized with the program.

U.3.10 MPEG-H_3dAudio multi-stream and command descriptors

The following MPEG-H_3dAudio descriptors may be present in the AF descriptor field:

AF_MPEG-H_3dAudio_multi-stream_descriptor() – The syntax and semantics of this descriptor is identical to the MPEG-H_3dAudio_multi-stream_descriptor() as defined in 2.6.114 and 2.6.115.

AF_MPEG-H_3dAudio_command_descriptor() – The syntax and semantics of this descriptor is identical to the MPEG-H_3dAudio_command_descriptor() as defined in 2.6.118.

NOTE – The descriptors MPEG-H_3dAudio_multi-stream_descriptor() as defined in 2.6.114 and MPEG-H_3dAudio_command_descriptor() as defined in 2.6.118 are optionally part of the PMT. In case of dynamic updates of those descriptors an update of the PMT would be required. In case PMT updates are not desirable, the corresponding AF_descriptors AF_MPEG-H_3dAudio_multi-stream_descriptor() respectively AF_MPEG-H_3dAudio_command_descriptor() can be distributed in the AF descriptor field.

U.3.11 Boundary descriptor

The boundary descriptor (see Table U.10) is used to indicate segment boundary information that can be used to support content partitioning requirements in different types of applications. When the descriptor is carried explicitly within an elementary stream, it indicates the frame accurate location of that boundary. Information carried in the boundary descriptor describes the boundary type, e.g., SAP type, partition and an optional sequence number of the partitioned virtual segment. When it is used together with the TEMI time descriptor, it may also be used to indicate a wall clock timestamp on the boundary point.

Partitions that are indicated by the boundary descriptor are a set of continuous segments divided by boundary descriptors within a media stream. A stream can be partitioned in several ways. For example, as shown in Figure U.1, Partition A corresponds to 2-second segment, while Partition B corresponds to 5-second segment, both share a common boundary point every 10 seconds.

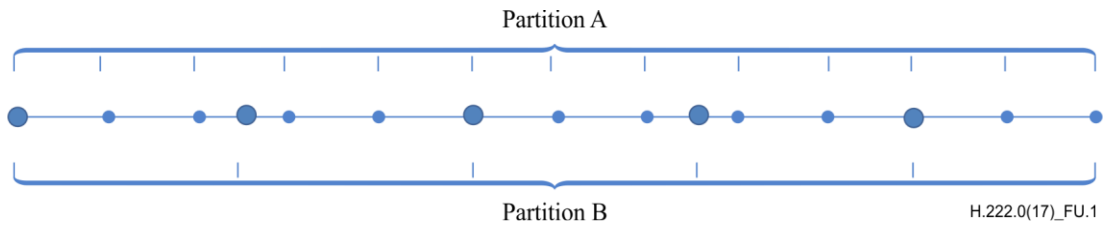


Figure U.1 – Stream partitioning into 2 and 5 second segments

Table U.10 – Boundary descriptor

Syntax	No. of bits	Format
boundary_descriptor() {		
af_descr_tag	**8**	**uimbsf**
af_descr_length	**8**	**uimbsf**
num_partitions_minus_1	**3**	**uimbsf**
SAP	**3**	**uimbsf**
concealment_flag	**1**	**bslbf**
reserved	**1**	**"0"**
for (i=0; i <= num_partitions_minus_1; i++) {		
partition_id	**3**	**uimbsf**
partition_info_flag	**1**	**bslbf**
reserved	**4**	**bslbf**
if (partition_info_flag == 1) {		
sequence_number_length_code (SNL)	**2**	**bslbf**
reserved	**6**	**uimbsf**
sequence_number	**see SNL**	**uismbf**
}		
}		
}		

U.3.12 Semantic definition of fields in boundary descriptor

SAP – SAP type, as defined ISO/IEC 14496-12.

num_partitions_minus_1 – Number of partitions to which this boundary belongs. By default (i.e., if this field is zero), each descriptor corresponds to a single partition.

partition_id – partition identifier

partition_info_flag – if set to 1 and SNL is not equal to zero, then extended partition information is available. If partition_info_flag is set to 1 and SNL is equal to zero, then this is reserved for future extension.

concealment_flag – Indicates if the source stream may have lost a frame that was at a boundary point. If flag is set to 1, indicates this is a repair boundary, inserted due to a previous lost boundary point frame. If set to 0, indicates no repair was done to the stream. "Repair" is meant to create a boundary point where it may have been skipped due to a lost boundary point frame.

sequence_number_length_code – Specifies length of the sequence_number field. The semantics for the width values are given in Table U.11.

Table U.11 – sequence_number_length_code interpretation

sequence_number_length_code	length, bits
0	0
1	16
2	32
3	64

sequence_number – Field providing a unique identifier for a segment within the context of this partition and multiplex. Length of this field is defined in sequence_number_length_code. If the sequence_number_length_code is 0, sequence_number field does not appear.

U.3.13 Labelling Descriptor

The Labelling Descriptor (see Table U.12) is used to carry one or more labels. When the descriptor is carried explicitly within an elementary system, it indicates a frame accurate location for the label or set of labels as a notation to the virtual segment. When used together with a boundary descriptor, it can note the start of an ad, program, chapter, or multi-period asset and carry identifiers (e.g., EIDR, or user defined) in the stream.

Table U.12 – Labelling Descriptor

Syntax	No. of bits	Format
labeling_descriptor() {		
af_descr_tag	**8**	**uimbsf**
af_descr_length	**8**	**uimbsf**
l = 0		
while (l < af_descr_length) {		
label_length_code		
label_type	**3**	**bslbf**
l += 2	**13**	**bslbf**
if (label_length_code == 7) {		
label_length		
l++	**8**	**uimbsf**
}		
label_bytes		
l += N	**N*8**	**bslbf**
}		
}		

NOTE – In Table U.12, the value of N is derived from label_length_code and label_length (if present), as described in U.3.14. For example, N would be 20 when label_length_code value is 6, and 7 when label_length_code and label_length both have value of 7.

U.3.14 Semantic definition of fields in labelling descriptor

label_length_code – Length of the label field in bytes, as provided in Table U.13.

Table U.13 – label_length_code interpretation

Value	Length, bytes
0	0 (not present)
1	2
2	4
3	8
4	12
5	16
6	20
7	Explicit value provided in ***label_length***

label_type – Type of the label field, as provided in Table U.14.

Table U.14 – label_type values

Range	Definition
0x00 .. 0xFF	Reserved for MPEG standardization
0x100 .. 0x1FF	Value defined in ANSI/SCTE 35 Table 8-8 (segmentation_type_id) + 0x100
0x200 .. 0x2FF	Value defined in ANSI/SCTE 35 Table 8-7 (upid_type_id) + 0x200
0x300 .. 0xFFF	Reserved for MPEG standardization
0x1000 .. 0x1FFF	User Private types

label_length – Length, in bytes, of the label field (i.e., number of label_bytes).

label_bytes – Bytes carrying the label. Length N is defined by label_length_code or label_length. Interpretation of these bytes depends on label types.

NOTE – If label_length_code is zero, this allows for label_type to exist with no label_bytes. This can serve as a marker type to the indicated PES frame.

Annex V

Transport of layered HEVC (MV-HEVC, SHVC)

(This annex does not form an integral part of this Recommendation | International Standard.)

V.1 Introduction

Transport of layered HEVC in MPEG-2 systems differs from earlier approaches in some respects, because the individual layers of an MV-HEVC or SHVC bitstream comply with certain profile, tier, and level restrictions depending on their combination within an output layer set, and there is no profile, tier, and level that applies to the output layer set as a whole, as had been for earlier layered video coding standards like SVC or MVC. In addition to this, layered HEVC can provide buffer models for individual layers, which had not been the case for previous scalable or multiview coding standards. The chosen approach for transport of layered HEVC in MPEG-2 systems allows sending at most one layer in one elementary stream, where the separate transport of temporal sub-layers of a single layer is supported. This allows alignment of the underlying HRD buffer model with the T-STD buffer model of systems.

V.2 Terminology

Figures V.1 and V.2 illustrate the terminology used for HEVC temporal sub-bitstreams and HEVC temporal video subsets of HEVC version 1. In Figure V.1, the horizontal axis indicates the different temporal sub-layers. Here, the value of the syntax element nuh_layer_id of all NAL units of all HEVC video sub-bitstreams is 0.

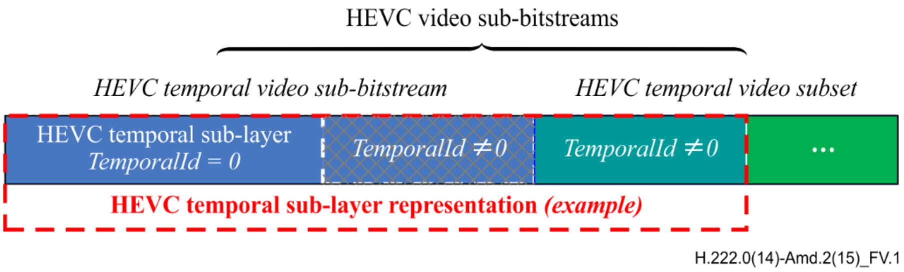


Figure V.1 – Terminology used for HEVC streams conforming to profiles defined in Annex A of ITU-T H.265 (2013) | ISO/IEC 23008-2:2013

Figure V.2 shows the additional layers introduced by the MV HEVC and SHVC extensions in the vertical dimension. In addition to the temporal layers, different values of the syntax element nuh_layer_id are allowed. Similar to Figure V.1, there are temporal video sub-bitstreams, which include the temporal layer with TemporalId equal to 0, and temporal video subsets, which do not. The HEVC base sub-partition contains data with LayerId 0 only, and is at the same time a temporal video sub-bitstream because it includes TemporalId 0. Furthermore, it is a conforming bitstream all by itself and can be independently decoded.

In general, an HEVC enhancement sub-partition can include an HEVC temporal video sub-bitstream with one layer or an HEVC temporal video subset with one layer as shown in Figure V.3. L_i, L_j and L_k stand for increasing values of nuh_layer_id larger than 0.

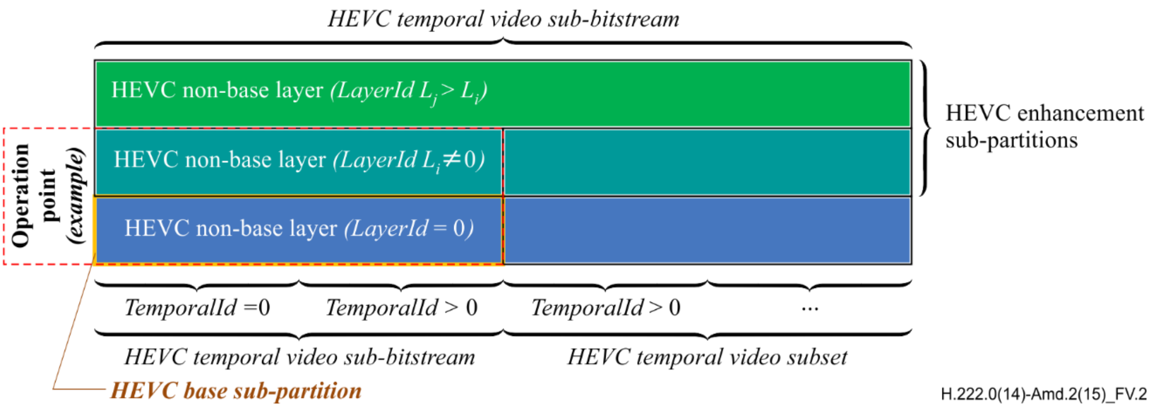


Figure V.2 – Terminology used for HEVC streams conforming to profiles defined in Annexes G and H

An elementary stream consists of an HEVC sub-partition for which one of the following is true:

- It is a conforming bitstream all by itself and can be independently decoded, i.e., it is an HEVC base sub-partition including all coded data with LayerId equal to 0 and TemporalId equal to 0;
- It enhances another elementary stream by adding one or more temporal sub-layers;
- It adds one layer identified by a value of nuh_layer_id > 0.

V.3 Examples

An example of an MV HEVC video stream is shown in Figure V.3. In this example, the base layer contains the central view (LayerId equal to 0) and two more views (left and right) are encoded as further layers with LayerIds equal to 1 and 2, respectively. For this example, the assumption is made that there are coding dependencies between left and central and between right and central only. Each layer is encoded in two temporal levels. In a straight-forward partitioning, all sub-partitions contain a temporal sub-bitstream or subset of a single layer. All possible operation points would be:

1. Single central view @30 fps (HEVC version 1);
2. Single central view @60 fps (HEVC version 1);
3. Left and central view @30 fps;
4. Left and central view @60 fps;
5. Right and central view @30 fps;
6. Right and central view @60 fps;
7. Left and right view @30 fps (central used for reference, but not output);
8. Left and right view @60 fps (central used for reference, but not output);
9. Left, right and central view @30 fps;
10. Left, right and central view @60 fps.

However, a service provider could decide that only a subset of these operation points should be signalled to the consumer, e.g., 1, 2, 3, 4, 7, 10.

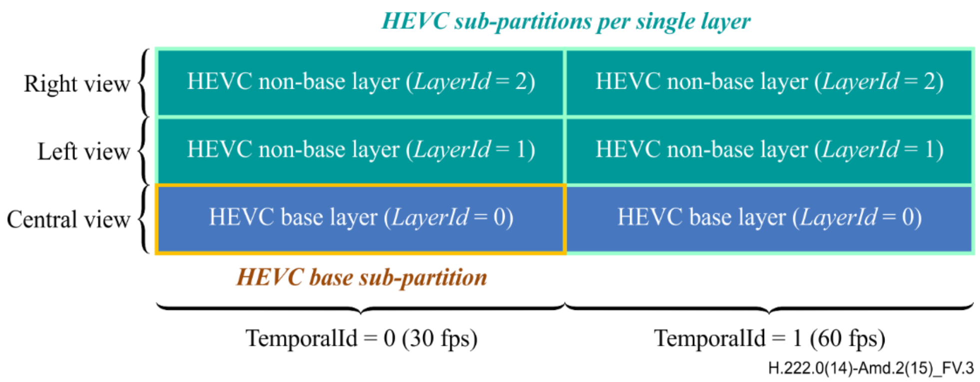


Figure V.3 – Example partitioning of MV-HEVC

In the example of Figure V.4, elementary streams with stream_type value 0x28 and 0x29 occur twice. The streams with stream_type 0x29 have two direct dependencies, thus an HEVC hierarchy extension descriptor is included in the program map table for both streams with stream_type 0x29. In contrast to that, one hierarchy descriptor is included in the program map table for the streams with stream_type0x24 and 0x25. Both streams with stream_type 0x28 have only one direct dependency, which could be signalled with a hierarchy descriptor; however, for this stream type, the use of the HEVC hierarchy extension descriptor is mandated. Even though all dependencies for the set of operations points shown in Figure V.4 could be signalled using a set of hierarchy descriptors and HEVC hierarchy extension descriptors, the HEVC operation point descriptor is mandated. In order to signal dependencies and profile/tier/level for all 10 operation points listed above, the HEVC operation point descriptor is needed.

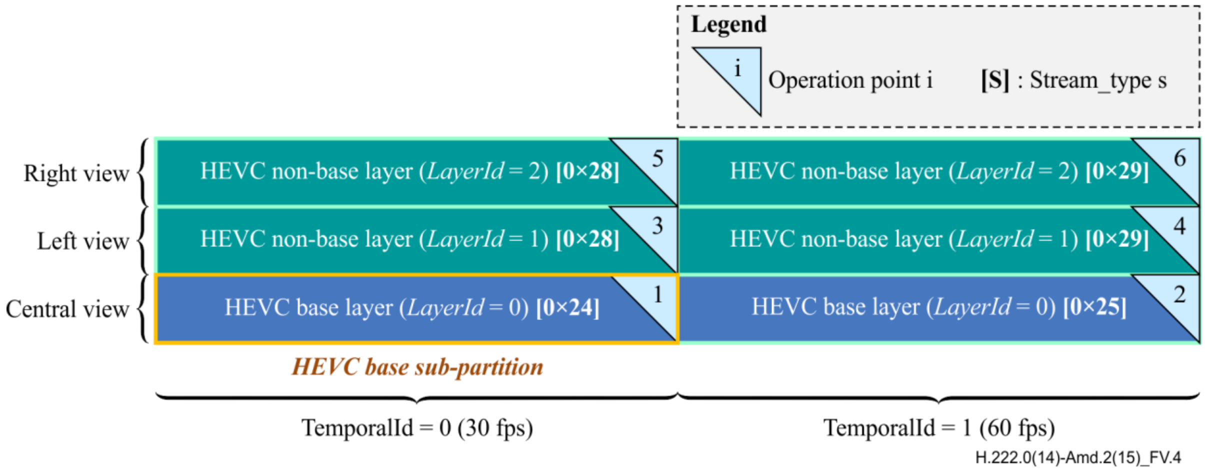


Figure V.4 – Example for layered transport of an MV-HEVC video stream

Figure V.5 shows an example for an SHVC video stream with five possible operation points, indicated by numbered triangles in the upper right corner of each box representing a sub-partition corresponding to an elementary stream. The lowest box represents the HEVC base layer. Further HEVC layers with values of the syntax element nuh_layer_id greater than 0 are shown in the vertical dimension. Note that in this example the frame rate enhancement through LayerId 2 does not add a temporal level, but uses TemporalId 0 again. In this case, dependencies between the streams are straight-forward. Each elementary stream depends on the elementary streams corresponding to operation points with lower numbers. It is sufficient to signal the dependency to the elementary streams corresponding to the operation point OP_{i-1} for the elementary streams corresponding to operation point OP_i. In total, four dependencies would be signalled:

- For OP2: Elementary stream corresponding to OP2 depends on the elementary stream corresponding to OP1;
- For OP3: Elementary stream corresponding to OP3 depends on the elementary stream corresponding to OP2;
- For OP4: Elementary stream corresponding to OP4 depends on the elementary stream corresponding to OP3;
- For OP5: Elementary stream corresponding to OP5 depends on the elementary stream corresponding to OP4.

In this example, indirect dependencies can easily be inferred by recursively resolving the dependencies. However, the HEVC operation point descriptor is mandated.

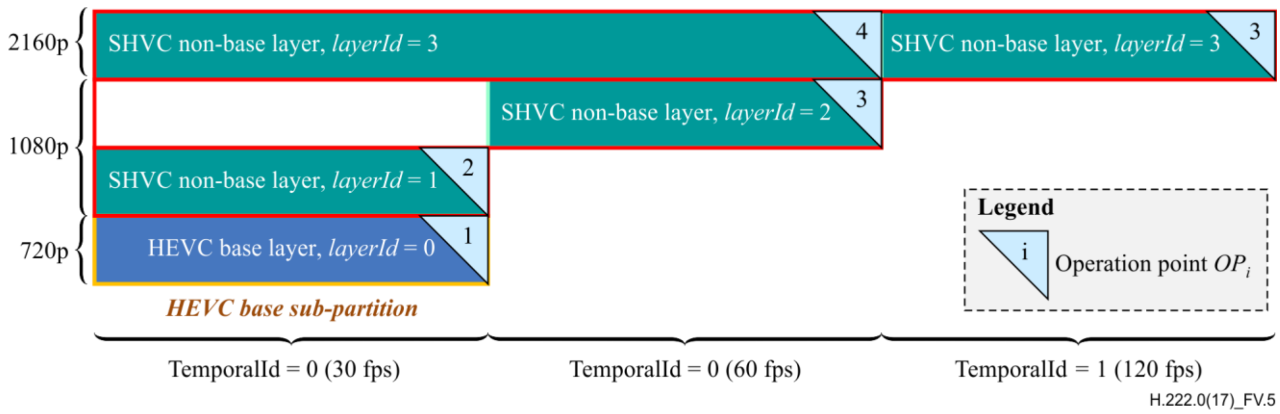


Figure V.5 – Example for layered transport of an SHVC video stream

Bibliography

– IETF RFC 6381 (2011), *The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types*.

– IETF RFC 3550 (2003), *RTP: A Transport Protocol for Real-Time Applications*.

– ISO/IEC 14496-18:2004, *Information technology – Coding of audio-visual objects – Part 18: Font compression and streaming*.

– ISO/IEC 23001-9:2014, *Information technology – MPEG systems technologies – Part 9: Common encryption of MPEG-2 transport streams*.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	**Audiovisual and multimedia systems**
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling, and associated measurements and tests
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2018